ECHO Reference Client Developer’s Guide Version 1.1

Table of Contents

1 Introduction

The ECHO Reference Client is a Java-based web application that interacts with NASA's Earth Observing System Clearinghouse (ECHO) to demonstrate an implementation of the following basic client operations:

- Login/logout
- Discover data providers
- Query Collections and Granules
- Order items
- Check order status

This application is distributed with the source code and is intended to be used as a learning resource by anyone developing software clients for ECHO. While the software is a fully working client, it is not meant to be an *operational* ECHO client. The user interface is very plain and the system supports only a small subset of the full ECHO functionality. More information on ECHO and client development is available from the project website: http://www.echo.nasa.gov.

This guide is divided into three major sections: installation and configuration of the client, a guided tour of its operation, and an explanation of the design of the application. A short section with suggestions for new client development concludes the document.

2 Installation & Configuration

2.1 Requirements

The following components are required by the ECHO Reference Client and should be available on the system where the reference client will be installed:

- Java 2 Platform, Standard Edition 1.4.x or later (e.g. JDK 5.0)
    - http://java.sun.com
- Apache Tomcat 5.5.9 or later
    - http://jakarta.apache.org/tomcat
- Apache Ant 1.5 or later
    - http://ant.apache.org

Note that since the Reference Client is a Java web application, it may be installed and run from any Java Web Server compliant with the *Servlet 2.2* (or later) and *JavaServer Pages 1.1* (or later) specifications. For the purposes of this guide, however, we use the Apache Tomcat web server because it is a widely documented and used application available for all platforms supporting the Java environment.

2.2 The Reference Client Distribution File

The ECHO Reference Client is distributed as a zip archive (`echorefclient-X.X.zip`--where "X.X" refers to the version of the Reference Client). Unzip the file to a local directory system. The directory structure shown in Table 1 will be created.

Table 1 - Directory Structure

`/echorefclient-X.X`	The root of the application distribution. This directory contains a script to build the application.
`/echorefclient-X.X/assemble`	Contains the Web Archive (WAR) file.
`/echorefclient-X.X/doc`	Contains the project documentation, including this guide and the Javadoc APIs. A sample log file is also provided.
`/echorefclient-X.X/source`	The root of the source code tree of the application. Also holds the logger configuration file.
`/echorefclient-X.X/webroot`	The web root of the assembled application. This contains the JavaServer Pages (JSPs), the Servlet WEB-INF, classes, and lib directories. The `web.xml` descriptor file is also stored here.

2.3 Building the Reference Client Application

The ECHO Reference Client uses the Apache Ant Java-based build tool to perform the compilation of the source code, bundle the files into a Java Web Archive (WAR), and generate the Javadocs.

The distribution file includes a pre-built WAR file for the application (in the `assemble` directory) and pre-generated Javadocs (in the `doc` directory).

If you make changes to the distribution source or configuration, you will need to use the Ant build file, `build.xml`, located at the root of the application distribution, to perform one of the following tasks:

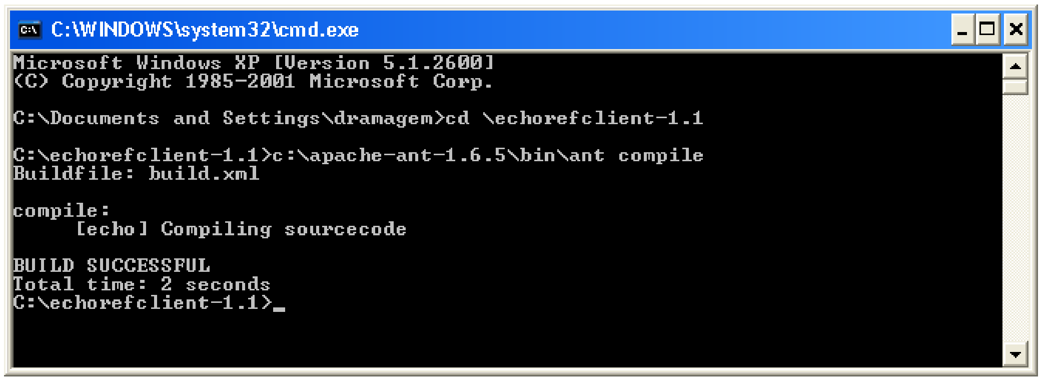
- **`compile`** – The default target. Compiles the source code into the `/echorefclient-X.X/webroot/WEB-INF/classes` directory.

- **`war`** – Creates (and overwrites) the `echorefclient.war` file and saves it in the `/echorefclient-X.X/assemble` directory. Will execute the `compile` script prior to performing this operation.

- **`javadoc`** – Creates the API documentation for the source code and saves it in the `/echorefclient-X.X/doc` directory.

- **`clean`** – Removes the contents of the `/echorefclient-X.X/webroot/WEB-INF/classes` and `/echorefclient-X.X/assemble` directories.

To execute any of these tasks, open a command shell to the root directory of the Reference Client distribution and call the `ant` executable (located in the `bin` directory of your Ant installation directory) followed by the target name, as shown in Figure 1.



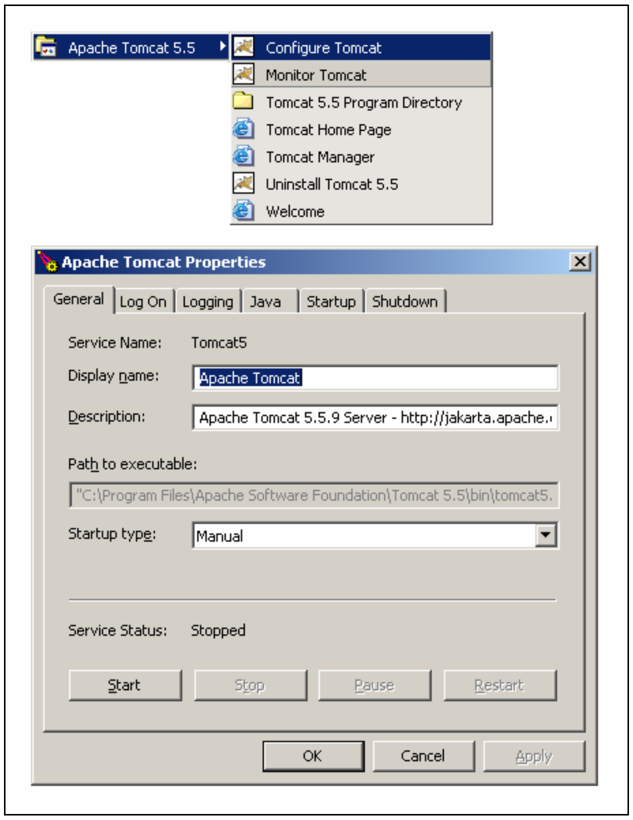
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\dramagem>cd \echorefclient-1.1

C:\echorefclient-1.1>c:\apache-ant-1.6.5\bin\ant compile
Buildfile: build.xml

compile:
     [echo] Compiling sourcecode

BUILD SUCCESSFUL
Total time: 2 seconds
C:\echorefclient-1.1>_
```

Figure 1 - Executing the Ant Build

2.4 *Running the Reference Client*

In order to run the Reference Client application, you will need to deploy the application on the Tomcat web server. In this section we will use the HTML version of the Tomcat Manager to install the Reference Client's Java Web Archive (WAR) file.

The Tomcat server needs to be started as per the instructions for your operating system. In the Microsoft Windows installation, a *program group* is created for Apache Tomcat 5.5 and a "Configure Tomcat" application is provided to start, stop and set various server settings. Refer to the documentation of your Tomcat distribution for the details on operating the server.



Service Name: Tomcat5

Display name: Apache Tomcat

Description: Apache Tomcat 5.5.9 Server - http://jakarta.apache.

Path to executable: "C:\Program Files\Apache Software Foundation\Tomcat 5.5\bin\tomcat5.

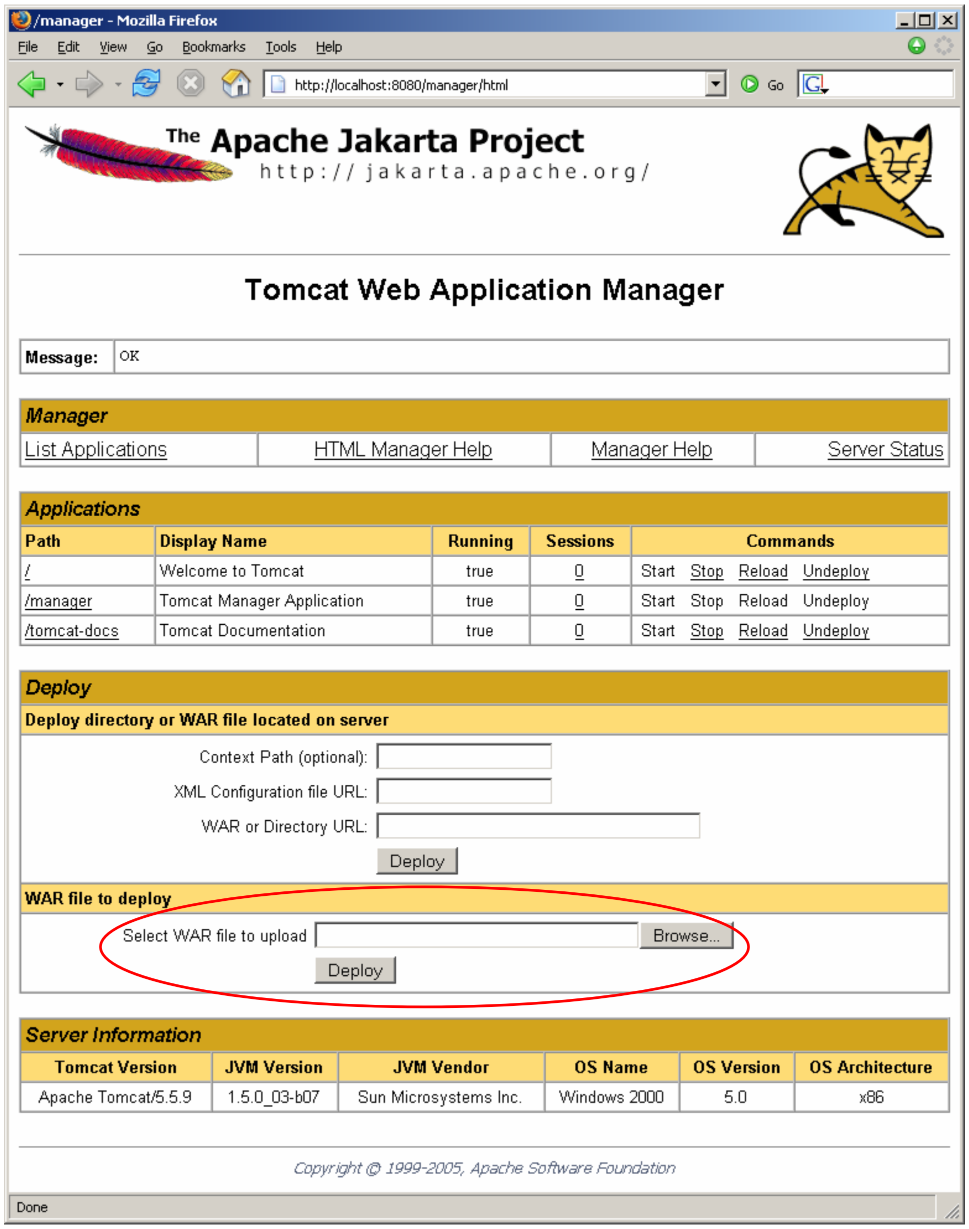
Startup type: Manual

Service Status: Stopped

Figure 2 - Tomcat on Microsoft Windows

After starting the Tomcat server, open a browser and access the appropriate URL for the Tomcat Manager (typically http://localhost:8080/manager/html).

In the section "WAR file to deploy" (Figure 3) enter (or browse for) the location of the `echorefclient.war` file (inside the `assemble` directory). Click on the *deploy* button. This will cause the application to be installed on the server.


/manager - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/manager/html
Go
The Apache Jakarta Project
http://jakarta.apache.org/

Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy

Deploy

Deploy directory or WAR file located on server

Context Path (optional):
XML Configuration file URL:
WAR or Directory URL:
Deploy

WAR file to deploy

Select WAR file to upload Browse...
Deploy

Server Information

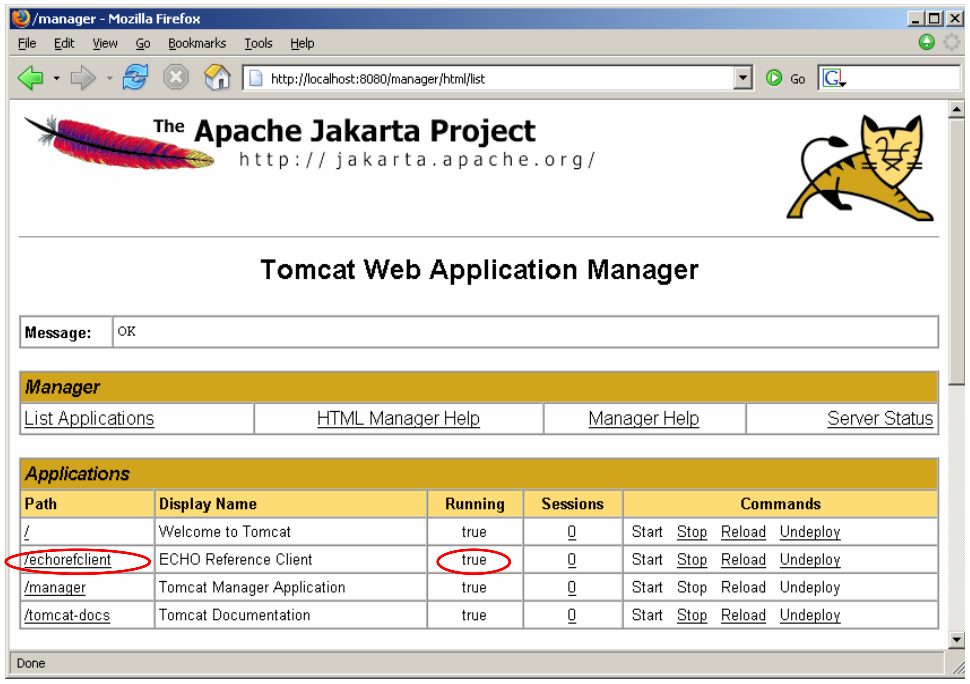
Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture
Apache Tomcat/5.5.9	1.5.0_03-b07	Sun Microsystems Inc.	Windows 2000	5.0	x86

Copyright © 1999-2005, Apache Software Foundation

Done

Figure 3 - Tomcat Manager – WAR File Deployment

Make sure that the Reference Client is running by examining the list of Applications and searching for an "echorefclient" entry as shown in Figure 4.


/manager - Mozilla Firefox

http://localhost:8080/manager/html/list

The Apache Jakarta Project
http://jakarta.apache.org/

Tomcat Web Application Manager

Message:	OK

Manager			
List Applications	HTML Manager Help	Manager Help	Server Status

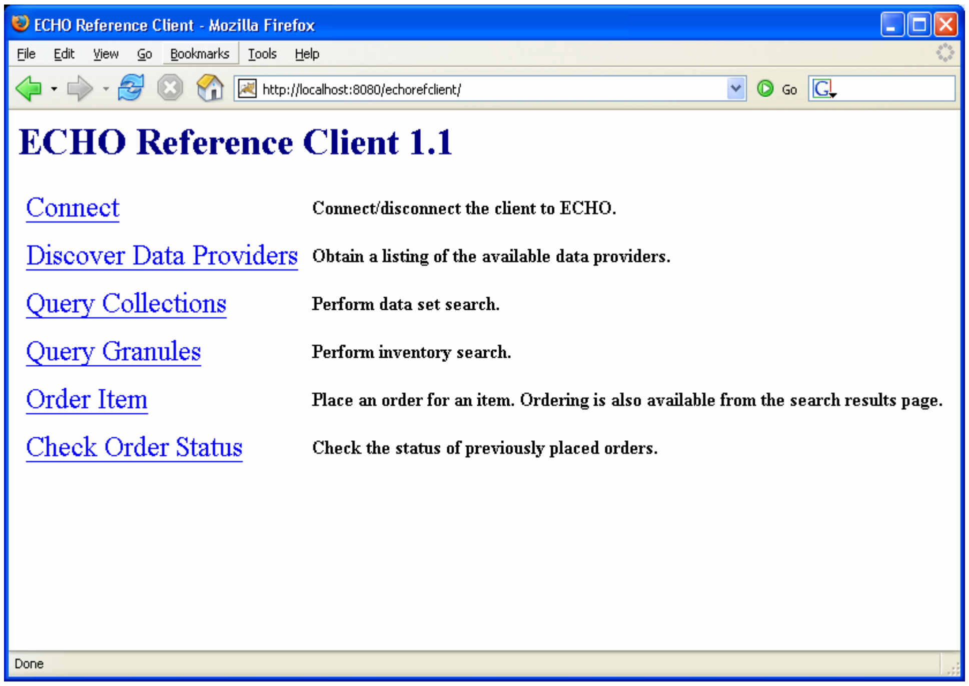
Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/echorefclient	ECHO Reference Client	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Undeploy

Done

Figure 4 - Tomcat Manager – List of Applications

To verify whether the deployment was successful, access the main page of the application: http://localhost:8080/echorefclient. If the installation was successful then you should be taken to the ECHO Reference Client Main screen (Figure 5).


ECHO Reference Client - Mozilla Firefox

http://localhost:8080/echorefclient/

ECHO Reference Client 1.1

Connect	Connect/disconnect the client to ECHO.
Discover Data Providers	Obtain a listing of the available data providers.
Query Collections	Perform data set search.
Query Granules	Perform inventory search.
Order Item	Place an order for an item. Ordering is also available from the search results page.
Check Order Status	Check the status of previously placed orders.

Done

Figure 5 - ECHO Reference Client Main Screen

In the case of any installation or deployment errors, Tomcat will display a web page with information about the problem(s) encountered. Tomcat configuration and installation is beyond the scope of this guide, so you will need to consult the Tomcat documentation for assistance on resolving such issues.

2.5 Logging

The ECHO Reference Client interacts with services provided by ECHO. You can monitor the XML messages that are exchanged between the Client and ECHO by examining the file `echorefclient.log` that is created (and by default appended—not overwritten—with newer information) in the root of the deployed web application (note that the log file is not initially created until a connection to ECHO is started). The log can be accessed directly via the web browser, at: http://localhost:8080/echorefclient/echorefclient.log.

The level of verbosity that is pre-configured with the Reference Client distribution should be adequate for most users, but may be modified as described in the rest of this section. A sample log file is provided in the `doc` directory (`sample-echorefclient.log`). We suggest you examine the log first before making changes. If you find it sufficiently detailed then you may wish to skip this section.

The Reference Client uses the standard Java 1.4 Logging API[1] and the level and detail of the log messages may be modified by two means:

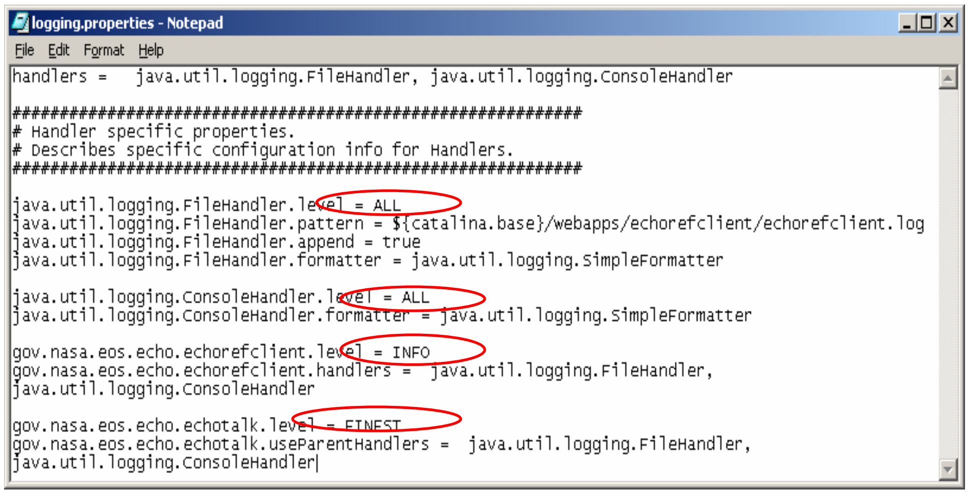
1. Changing the deployed `logging.properties` file (located under the Tomcat `webapps/echorefclient/WEB-INF/classes`) and restarting the Reference Client application (e.g., "stop" and "start" in the Tomcat HTML Manager—refer to Figure 4).

2. Changing the `logging.properties` file located in the `source` directory, re-building the WAR file (i.e., running "ant war"—see Section 2.3), and re-deploying the new WAR (this procedure was described in Section 2.4).

In either case, to change the verbosity of the logging it is necessary to modify the "level" properties defined in the `logging.properties` file. This file (Figure 6) specifies both file and console (i.e., *standard output*) as output streams for the log messages.

The location of the log file is set by the property `org.apache.juli.FileHandler.directory` (by default this property is set to the Tomcat "`webapps/echorefclient`" directory). The related `java.util.logging.FileHandler.append` property (set to `true` by default) determines if this file gets appended or overwritten upon every restart of the application or Tomcat server.

[1] http://java.sun.com/j2se/1.4.2/docs/guide/util/logging



```
handlers =   java.util.logging.FileHandler, java.util.logging.ConsoleHandler

############################################################
# Handler specific properties.
# Describes specific configuration info for Handlers.
############################################################

java.util.logging.FileHandler.level = ALL
java.util.logging.FileHandler.pattern = ${catalina.base}/webapps/echorefclient/echorefclient.log
java.util.logging.FileHandler.append = true
java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter

java.util.logging.ConsoleHandler.level = ALL
java.util.logging.ConsoleHandler.formatter = java.util.logging.SimpleFormatter

gov.nasa.eos.echo.echorefclient.level = INFO
gov.nasa.eos.echo.echorefclient.handlers =  java.util.logging.FileHandler,
java.util.logging.ConsoleHandler

gov.nasa.eos.echo.echotalk.level = FINEST
gov.nasa.eos.echo.echotalk.useParentHandlers =  java.util.logging.FileHandler,
java.util.logging.ConsoleHandler
```

Figure 6 - Logging Properties File – Relevant Property Values Highlighted

Both the file and console outputs have their log levels set to `ALL` by default but may be set instead to any of the following values:

- `ALL` - Log all messages generated by ECHO.
- `FINEST` - Log the XML responses *from* ECHO.
- `FINER` - Log the XML requests sent *to* ECHO
- `FINE` - Log the connection/disconnection information and general request/response details.
- `INFO` - Log the duration of ECHO transactions (the time it takes to receive the raw response from ECHO once the request has been sent).
- `SEVERE` - Log any errors that occur during communication with ECHO.

Note that the levels defined higher up in the list also encompass the levels beneath—for instance, FINE also covers the logs of INFO and SEVERE.

In addition to setting the log level of the outputs, the Java application's *package* log level also needs to be set, similar to the file and console outputs. For the ECHO Reference Client, only the primary packages should be set to fixed levels. This involves setting the properties:

- `gov.nasa.eos.echo.echorefclient.level = INFO`
- `gov.nasa.eos.echo.echotalk.level = FINEST`

By default, the *echorefclient* package is set to log level INFO, which will cause all exceptions generated by the application to be logged. Log messages generated by the *echotalk* package are set to log level FINEST by default, which will display the full XML exchange between the client and ECHO.

3 Guided Tour

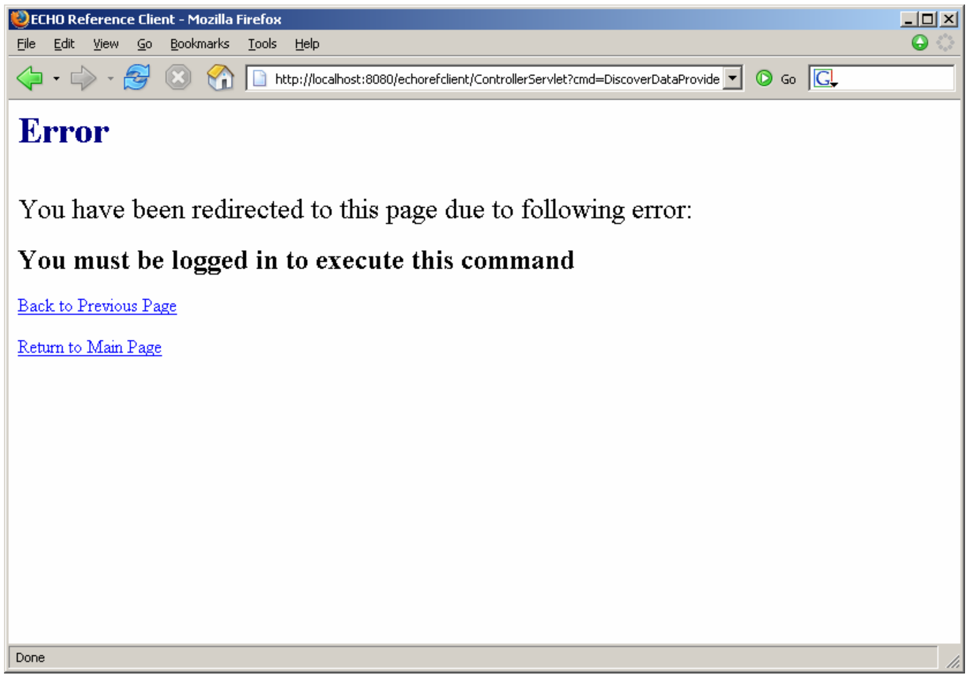
This section provides example usage of the ECHO Reference Client and provides helpful observations and tips along the way. Start by accessing the application (e.g., http://localhost:8080/echorefclient). The Main page should come up (Figure 8).

Behind-the-scenes Look

As you go through this tour you may want to monitor the application as described in section 2.5 (Logging). This will give you insight into the operation of the Reference Client and a look at the XML messages actually being exchanged with ECHO.

3.1 Reference Client Error Handling

If an error happens to occur while you're going through this tour of the Reference Client, an error page will be displayed with information about the cause (Figure 7). The nature of the error may be an application fault, a network-related connectivity problem (with the machine hosting the Client), or an ECHO notification of incorrect or invalid data being submitted (e.g., entering letters for a numeric field).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=DiscoverDataProvide
Go
Done

Error

You have been redirected to this page due to following error:

You must be logged in to execute this command

Back to Previous Page

Return to Main Page

Figure 7 - Error Page – Sample Application Fault

When an error occurs, try to re-execute your last action by returning to the previous page (selecting the "Back to Previous Page" link or the back button on your web browser). Make any necessary corrections to form fields and re-submit the information.

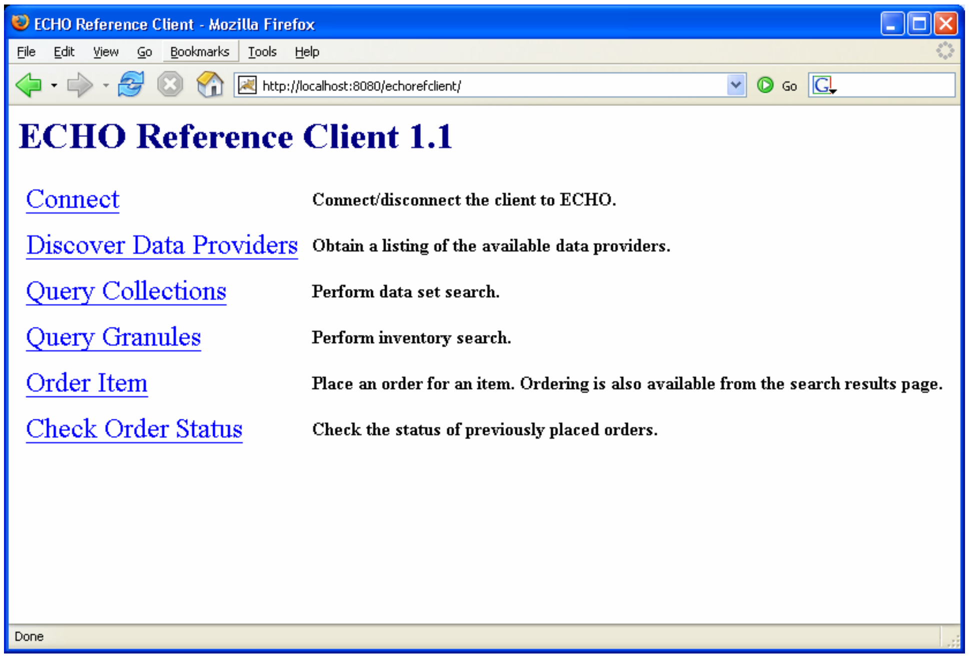
Session Timeout

If you leave the application inactive for too long (more than 60 minutes) after connecting you may get a "Your session has expired" error message indicating that you will need to log in again. This is an application timeout, not an ECHO server session expiration.

The application timeout value is a Servlet container attribute that can be configured at the Java Web Server level or inside the application's `web.xml` file by using the `<session-timeout>` tag. Refer to your web server's documentation or the Java Servlet specifications for details.

3.2 Main Page

The Main page (Figure 8) provides links for the various functions of the application. Before accessing any of these functions, you must first tell the client whether to establish a connection with ECHO as a *non-registered* (i.e., "guest") user or use a *registered* ECHO user account.


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/
Go
Done

ECHO Reference Client 1.1

Connect	Connect/disconnect the client to ECHO.
Discover Data Providers	Obtain a listing of the available data providers.
Query Collections	Perform data set search.
Query Granules	Perform inventory search.
Order Item	Place an order for an item. Ordering is also available from the search results page.
Check Order Status	Check the status of previously placed orders.

Figure 8 - Main Page

The ECHO Session Manager & User System

All client interaction with ECHO occurs through the *ECHO Session Manager*, which exposes a SOAP interface with a *login* function to establish a user session (if the client chooses to connect to ECHO as a registered user). It is only upon the creation of the user session that additional requests can be performed by ECHO. The Reference Client *Login* function executes the Session Manager *login* method and provides appropriate user credentials.

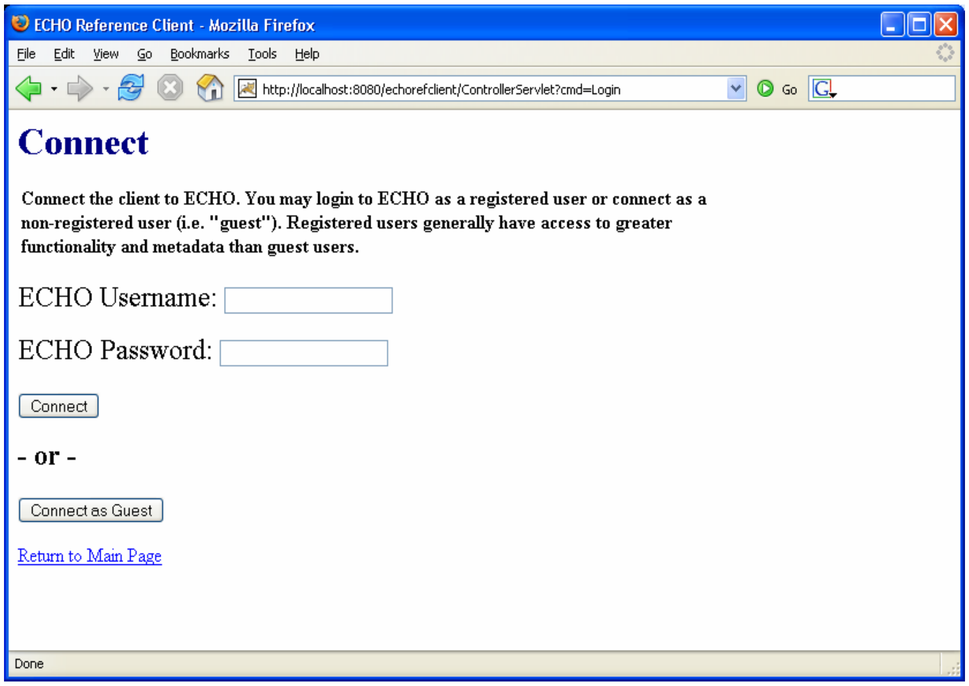
Note that the ECHO Reference Client does not provide any account registration services for

ECHO. Contact the ECHO Operations Team (http://www.echo.nasa.gov) to register for a user account. In general registered users have access to greater ECHO functionality and metadata than non-registered users.

For the Reference Client, all of the functions are available for both registered and non-registered ECHO users. However, a registered user will have access to a listing of his previously placed orders, whereas the non-registered will be required to enter an order number to obtain its status. Also, a registered user may have access to more metadata or may be able to order items that the non-registered user cannot.

3.3 Connection Page

Click on the *Connect* link. The Connection page is presented (Figure 9) and you are prompted to either provide ECHO user credentials or to continue as a "guest" user. To proceed without a registered user account, click on the *Connect as Guest* button.


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=Login
Go

Connect

Connect the client to ECHO. You may login to ECHO as a registered user or connect as a non-registered user (i.e. "guest"). Registered users generally have access to greater functionality and metadata than guest users.

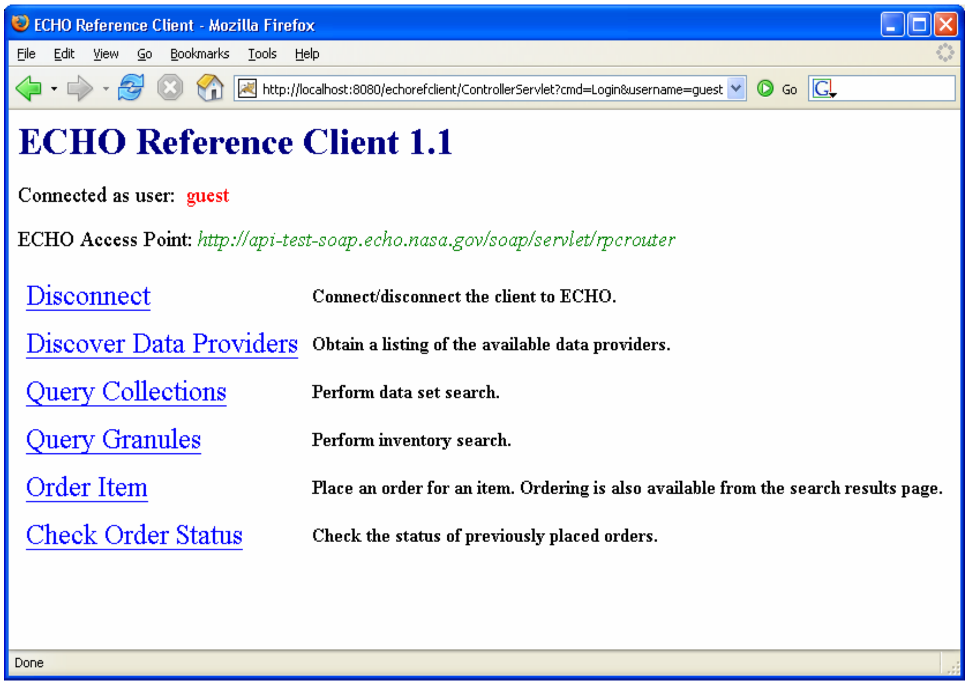
ECHO Username:

ECHO Password:

Connect
- or -
Connect as Guest
Return to Main Page
Done

Figure 9 - Connection Page

After connecting to ECHO you will be taken back to the Main page (Figure 10). Notice that now your user credential is displayed at the top, along with the URL of the ECHO server that the application is using.


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=Login&username=guest
Go
Done

ECHO Reference Client 1.1

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Disconnect	Connect/disconnect the client to ECHO.
Discover Data Providers	Obtain a listing of the available data providers.
Query Collections	Perform data set search.
Query Granules	Perform inventory search.
Order Item	Place an order for an item. Ordering is also available from the search results page.
Check Order Status	Check the status of previously placed orders.

Figure 10 - Main Page after Login

ECHO Test & Operational Servers

Access to ECHO is available via two URLs:

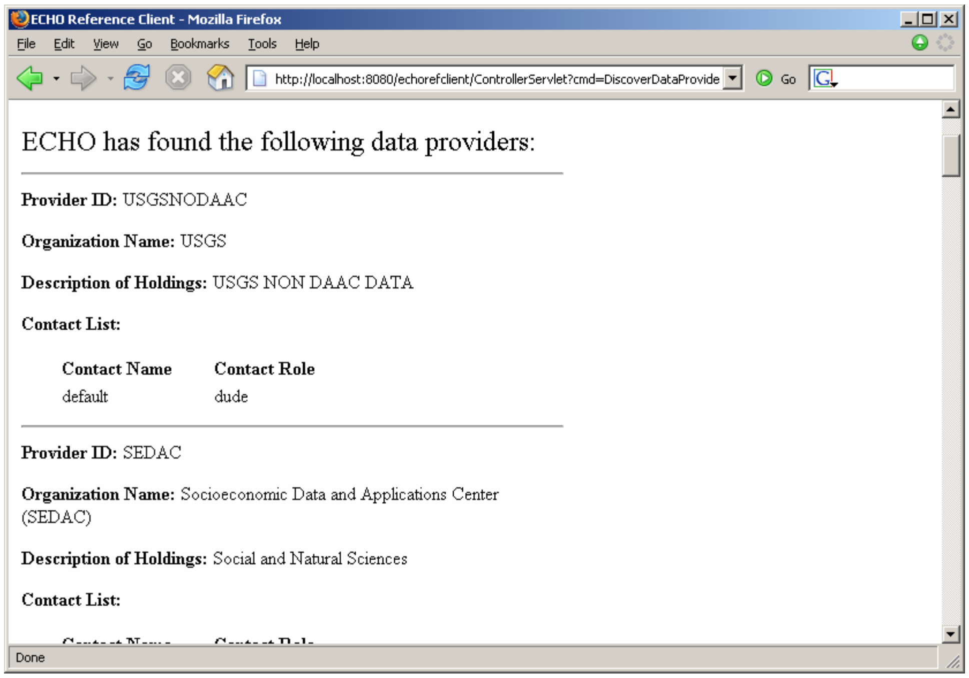
- ECHO API Test – http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter
- ECHO API – http://api-soap.echo.nasa.gov/soap/servlet/rpcrouter

The Reference Client is by default configured to access the Test system, which mimics the behavior of the operational system and serves as a testing ground for client development. The Reference Client can be set to use the operational system.

In order to change the server it is necessary to modify the `gov.nasa.eos.echo.echorefclient.Constants` class and change the value of the constant `URL_IN_USE`. Because this value is hard coded, it will be necessary to re-compile, re-build, and re-deploy the application (refer to directions in section 2 "Installation & Configuration").

3.4 Discover Data Providers

Once you've connected with ECHO you can perform the other functions listed. Let's begin with learning more about the Data Providers currently registered with ECHO by clicking on the *Discover Data Providers* link. A page with detailed information about the various ECHO Data Providers is displayed (Figure 11).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=DiscoverDataProvide
Go
Done

ECHO has found the following data providers:

Provider ID: USGSNODAAC

Organization Name: USGS

Description of Holdings: USGS NON DAAC DATA

Contact List:

Contact Name	Contact Role
default	dude

Provider ID: SEDAC

Organization Name: Socioeconomic Data and Applications Center (SEDAC)

Description of Holdings: Social and Natural Sciences

Contact List:

Figure 11 - Discover Data Providers

<u>Discover Data Providers</u>

The *Discover Data Providers* function makes use of:

- ECHO API Provider Profile Service
  - List All Providers Transaction
  - Present Provider Profile Transaction

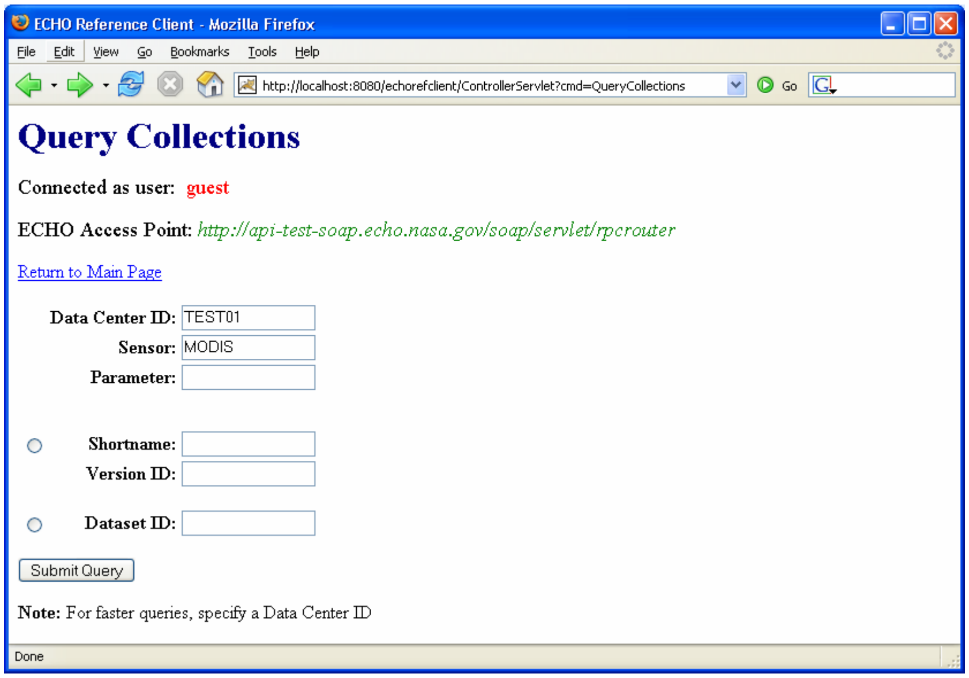
Some of the information displayed on the Data Provider information page may be incomplete. The Reference Client is set by default to work with the ECHO Test system ("api-soap-test"). This system may contain artificial information and metadata for testing purposes. Even in the Operational ECHO system ("api-soap"), it is up to the Data Partners to keep their information accurate and up-to-date.

Note also that the information that is displayed by the Reference Client will frequently be a subset of the full information that is available from ECHO. This is because the programming modules that comprise the Reference Client have been written to be selective about the information extracted from ECHO.

It is always a best practice to limit, when possible, the information you request from ECHO to the minimum required (some transactions allow you to specify the metadata to be returned—such as in the Catalog Service's Query Request). In doing this you will optimize performance of your client and minimize its use of ECHO's shared resources.

3.5 Query Collections

Now return to the Main page and click on the *Query Collections* link. A form is displayed with various search criteria. Filling the criteria fields with text narrows the search to be performed. Let's perform a search for Collections in the "TEST01" provider (one of the Provider IDs we previously obtained from the Discover Data Providers results). Fill in the *Data Center ID* field with this provider ID and set the *Sensor* value to "MODIS" (Figure 12). Press the *Submit Query* button.


ECHO Reference Client - Mozilla Firefox

http://localhost:8080/echorefclient/ControllerServlet?cmd=QueryCollections

Query Collections

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Return to Main Page

Data Center ID:	TEST01
Sensor:	MODIS
Parameter:	
○ Shortname:	
Version ID:	
○ Dataset ID:	

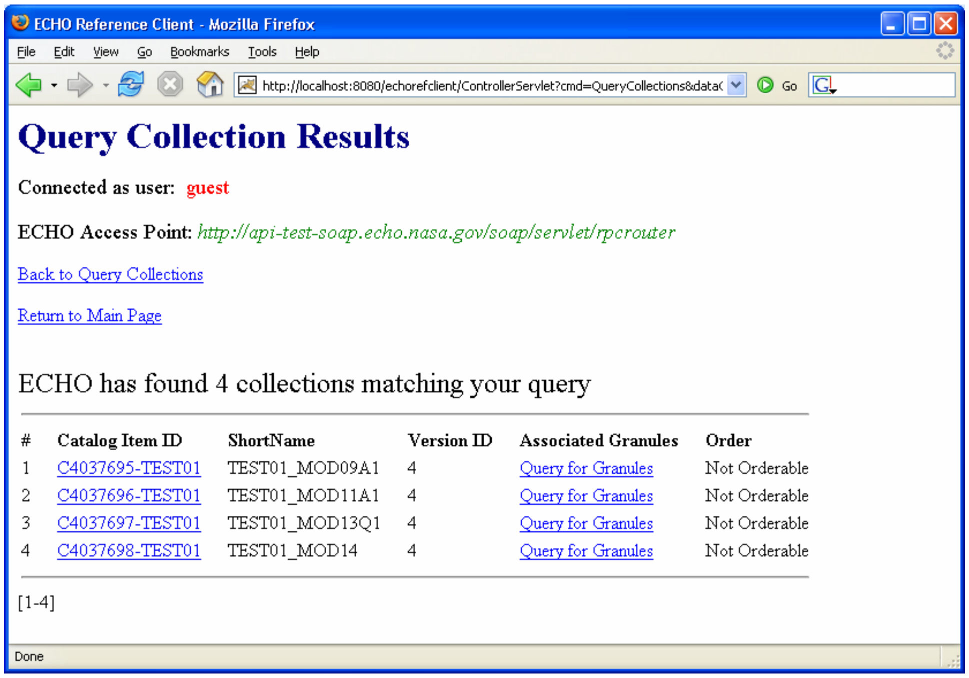
Submit Query

Note: For faster queries, specify a Data Center ID

Done

Figure 12 – Querying for MODIS Collections from the TEST01 Provider

A results page is displayed (Figure 13). Depending on the number of hits, you can iterate through the result set by selecting the *grouped range* of results (e.g., "1-10," "11-20," etc.) and/or the *Next* and *Previous* links.


ECHO Reference Client - Mozilla Firefox

http://localhost:8080/echorefclient/ControllerServlet?cmd=QueryCollections&data(

Query Collection Results

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Back to Query Collections

Return to Main Page

ECHO has found 4 collections matching your query

#	Catalog Item ID	ShortName	Version ID	Associated Granules	Order
1	C4037695-TEST01	TEST01_MOD09A1	4	Query for Granules	Not Orderable
2	C4037696-TEST01	TEST01_MOD11A1	4	Query for Granules	Not Orderable
3	C4037697-TEST01	TEST01_MOD13Q1	4	Query for Granules	Not Orderable
4	C4037698-TEST01	TEST01_MOD14	4	Query for Granules	Not Orderable

[1-4]

Done

Figure 13 – Query Collection Results Page

Query Collections

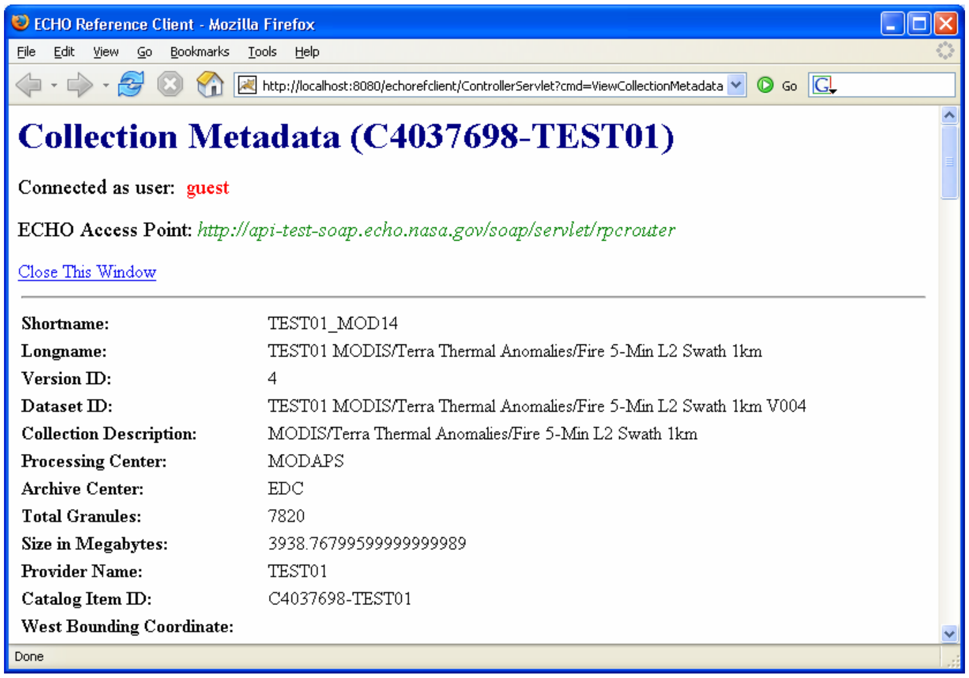
The *Query Collections* function makes use of:

- ECHO API Catalog Service
  - Query Transaction
  - Present Transaction
- Independent Information System Alternative Query Language (IIMSAQL)

The ECHO IIMSAQL makes available nineteen different fields for querying for Collections. The Reference Client only makes use of a subset of these:

- **Data Center ID** – It is recommended that a Data Center ID be entered in order to improve performance of the results. Leaving this field blank will query across all registered data providers and may take a long time to complete.
- **Sensor** – The short name of a sensor (e.g., "MODIS", "AVHRR", etc.).
- **Parameter** – This Collection condition references the "VariableKeyword" elements of the "DisciplineTopicParameters" field in ECHO's Collection metadata ingest DTD. Examples of the values for this field include: "Fire Occurrence", "Land Surface Temperature", "Leaf Characteristics", "Reflectance", "Terrain Elevation", and "Vegetation Index". The Reference Client uses input provided from the user to form a text pattern search (rather than exact value match) when querying ECHO.
- **Short Name & Version ID** – In Catalog Service Query transactions, Collections are uniquely identified by a Short Name plus Version ID (e.g., "TEST01_MOD09A1", version "4"), *or* by a unique Dataset ID (see below).
- **Dataset ID** – The unique dataset identifier (e.g., "TEST01 MODIS/Terra Surface Reflectance 8-Day L3 Global 500m SIN Grid V004").

Now click on the *ECHO Item ID* of an individual Collection result. A new window is displayed with detailed information about the Collection (Figure 14).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=ViewCollectionMetadata
Go
Done

Collection Metadata (C4037698-TEST01)

Connected as user: guest

ECHO Access Point: http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter

Close This Window

Shortname:	TEST01_MOD14
Longname:	TEST01 MODIS/Terra Thermal Anomalies/Fire 5-Min L2 Swath 1km
Version ID:	4
Dataset ID:	TEST01 MODIS/Terra Thermal Anomalies/Fire 5-Min L2 Swath 1km V004
Collection Description:	MODIS/Terra Thermal Anomalies/Fire 5-Min L2 Swath 1km
Processing Center:	MODAPS
Archive Center:	EDC
Total Granules:	7820
Size in Megabytes:	3938.76799599999999989
Provider Name:	TEST01
Catalog Item ID:	C4037698-TEST01
West Bounding Coordinate:	

Figure 14 - Collection Metadata Page

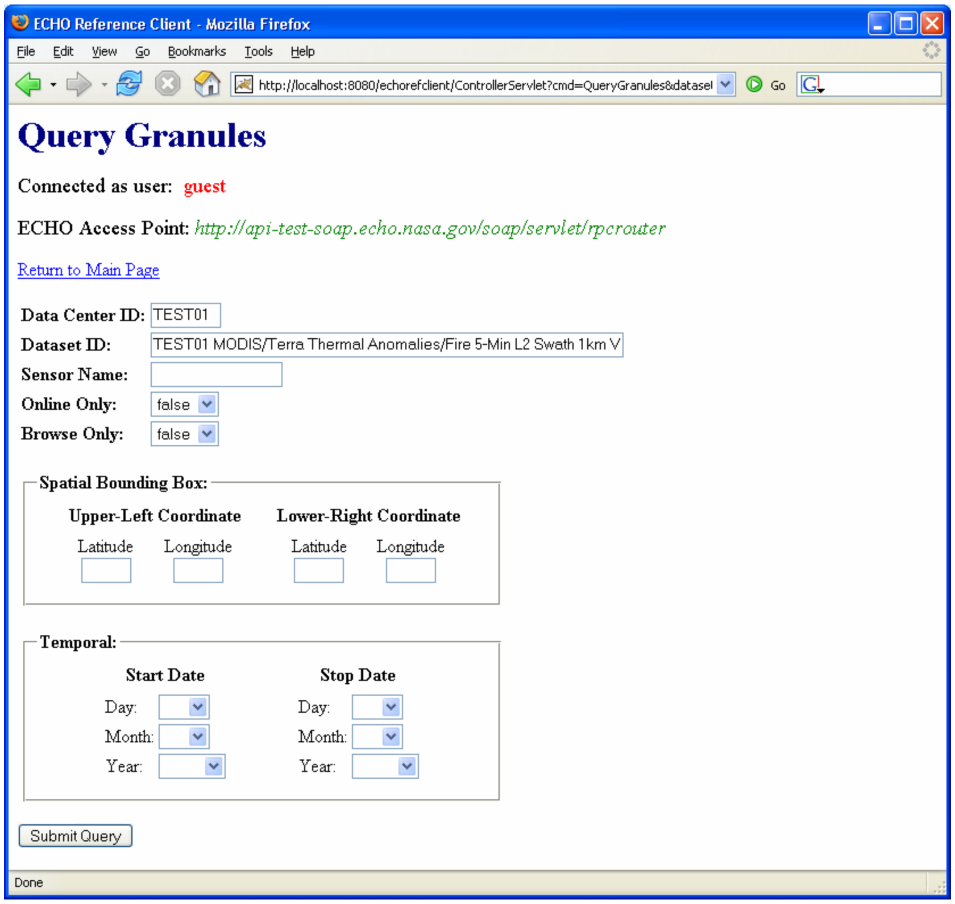
Collection Metadata

In the ECHO data model Collections are made up of dozens of metadata fields defined in http://api.echo.nasa.gov/echo/dtd/ECHOCollectionResults.dtd. The Reference Client only captures a small subset of these fields.

3.6 Querying Granules

Up to now we were able to find a set of MODIS Collections offered by the TEST01 provider. Now let's examine the set of Granules that actually make up one of these Collections. For this we will use the *Query Granules* function. There are two ways of performing this function: through the *Query Collection Results* page or directly from the *Main* page.

Let's try the first option. Close the *Collection Metadata* page (Figure 14). You should be back to the *Query Collection Results* page (Figure 13). Click on the *Query for Granules* link next to one of the Collections. This takes you to the *Query Granules* page (Figure 15). Note that the *Data Center ID* and *Dataset ID* fields are pre-filled with the information from the selected Collection. This page is the exact same page you will see if you click on the *Query Granules* link from the *Main* page.


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=QueryGranules&datasel
Go
Done

Query Granules

Connected as user: guest

ECHO Access Point: http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter

Return to Main Page

Data Center ID:	TEST01
Dataset ID:	TEST01 MODIS/Terra Thermal Anomalies/Fire 5-Min L2 Swath 1km V
Sensor Name:	
Online Only:	false
Browse Only:	false

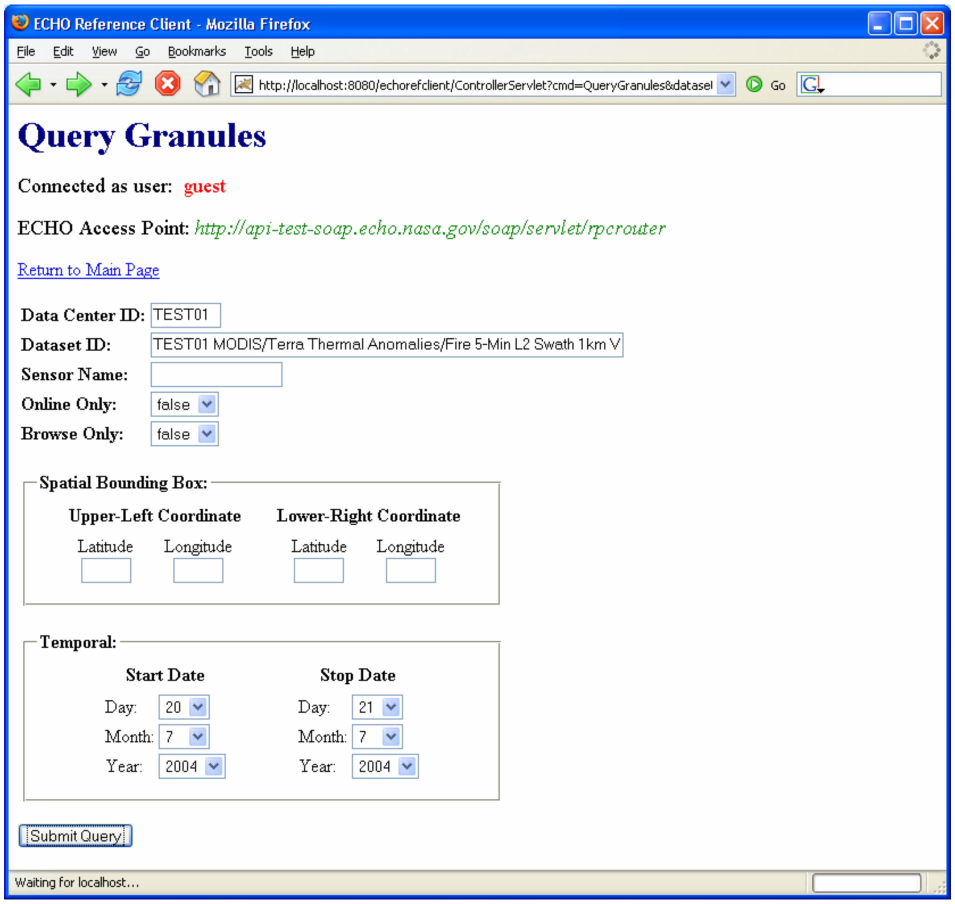
Spatial Bounding Box:
Upper-Left Coordinate: Latitude, Longitude
Lower-Right Coordinate: Latitude, Longitude

Temporal:
Start Date: Day, Month, Year
Stop Date: Day, Month, Year

Submit Query

Figure 15 - Query Granules

We have the option of imposing additional restrictions into the Granules retrieved from the Collection, such as spatial and temporal constraints. It is strongly recommended that the searches be constrained as much as possible because otherwise they can create very large result sets. So let’s add a temporal range to our search by inputting the start date of July 20th, 2004 and an end date of July 21st, 2004 (Figure 16).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Go

http://localhost:8080/echorefclient/ControllerServlet?cmd=QueryGranules&datasel

Query Granules

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Return to Main Page

Data Center ID:	TEST01
Dataset ID:	TEST01 MODIS/Terra Thermal Anomalies/Fire 5-Min L2 Swath 1km V
Sensor Name:	
Online Only:	false
Browse Only:	false

Spatial Bounding Box:

Upper-Left Coordinate		Lower-Right Coordinate	
Latitude	Longitude	Latitude	Longitude

Temporal:

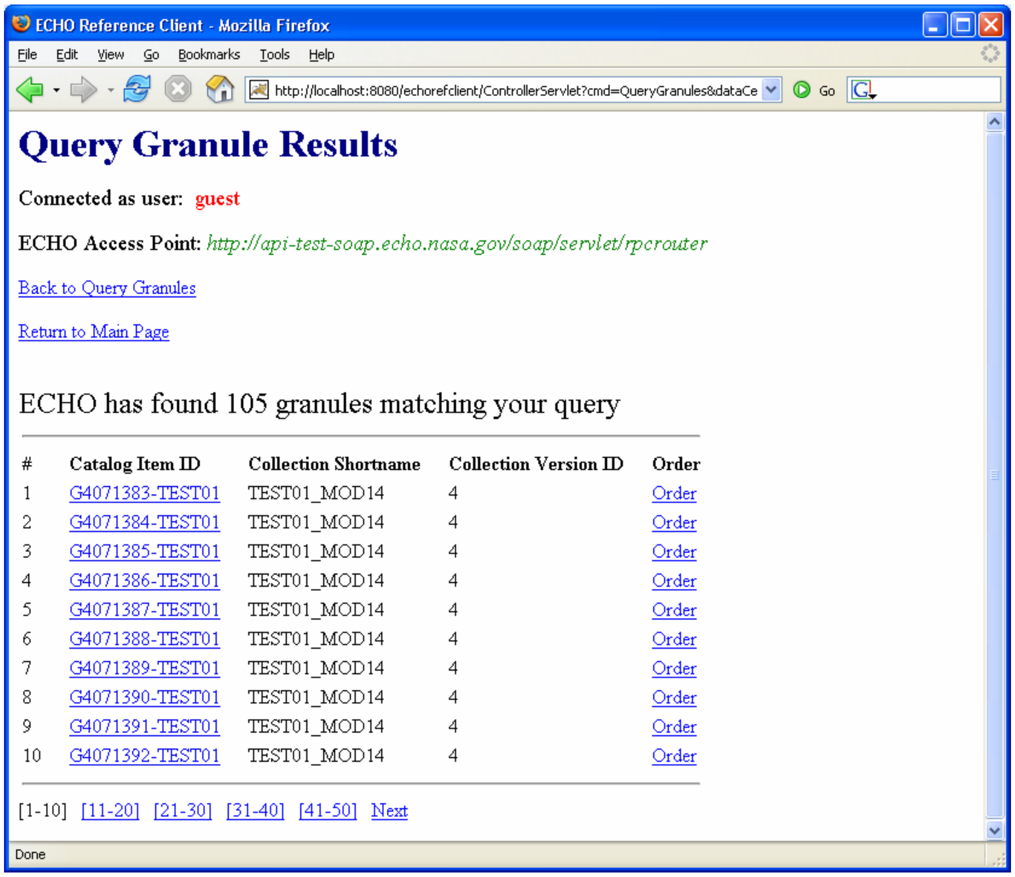
Start Date		Stop Date	
Day:	20	Day:	21
Month:	7	Month:	7
Year:	2004	Year:	2004

Submit Query

Waiting for localhost...

Figure 16 - Query Granules with Added Temporal Constraint

Click on the *Submit Query* button. The *Query Granules Results* page is returned (Figure 17).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=QueryGranules&dataCe
Go
Done

Query Granule Results

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Back to Query Granules

Return to Main Page

ECHO has found 105 granules matching your query

#	Catalog Item ID	Collection Shortname	Collection Version ID	Order
1	G4071383-TEST01	TEST01_MOD14	4	Order
2	G4071384-TEST01	TEST01_MOD14	4	Order
3	G4071385-TEST01	TEST01_MOD14	4	Order
4	G4071386-TEST01	TEST01_MOD14	4	Order
5	G4071387-TEST01	TEST01_MOD14	4	Order
6	G4071388-TEST01	TEST01_MOD14	4	Order
7	G4071389-TEST01	TEST01_MOD14	4	Order
8	G4071390-TEST01	TEST01_MOD14	4	Order
9	G4071391-TEST01	TEST01_MOD14	4	Order
10	G4071392-TEST01	TEST01_MOD14	4	Order

[1-10] [11-20] [21-30] [31-40] [41-50] Next

Figure 17 - Query Granules Results

Query Granules

Similar to the *Query Collections*, the *Query Granules* function makes use of:

- ECHO API Catalog Service
  - Query Transaction
  - Present Transaction
- Independent Information System Alternative Query Language (IIMSAQL)

The IIMSAQL makes available over fifteen different fields for querying Granules. The Reference Client only makes use of a subset of these:

- **Data Center ID** – It is recommended that a Data Center ID be entered in order to improve performance and the speed of the results. Leaving this field blank will query across all registered data providers and may take a long time to complete.
- **Dataset ID** – The unique identifier of a Collection. If this field is provided then the query will be performed just over the Collection.
- **Sensor Name** – The short name of a sensor (e.g. "MODIS", "AVHRR", etc.).
- **Online Only** – Searches for Granules that are marked by the Provider as being available online.
- **Browse Only** – Searches for Granules that are marked by the Provider as having browse images available online.
- **Spatial** – Constrains the search for Granules over a spatial region using geographical coordinates.
- **Temporal** – Constrains the search for Granules over a time period.

More on Temporal Searching in ECHO

Granule metadata can be searched in ECHO using constraints on several temporal parameters: observation or "acquisition" date, Provider Production Date, Provider Insert Date, ECHO Insert Date, and ECHO Last Update. The values for these parameters are stored in ECHO as Gregorian dates in GMT.

The temporal search implemented in the Query Granules function of the Reference Client relates to the `RangeDateTime` or `SingleDateTime` parameters supplied by Data Partners in their Granule metadata to characterize the observation date(s), with beginning and ending dates (day, month, and year) specified by the user. For the example shown in Figure 17, the message sent to ECHO will contain the following Granule temporal condition:

```
<granuleCondition>
  <temporal>
    <startDate><Date YYYY="2004" MM="07" DD="20"/></startDate>
    <stopDate><Date YYYY="2004" MM="07" DD="21"/></stopDate>
  </temporal>
</granuleCondition>
```

Greater time precision can be achieved by adding values for hour, minute, and second parameters to the `<Date>` specification. In addition, ECHO allows users to specify a periodic temporal range for Collection and Granule query. More information on temporal searching is provided in Section 6.1.3 of the ECHO User's Guide.

More on Spatial Searching in ECHO

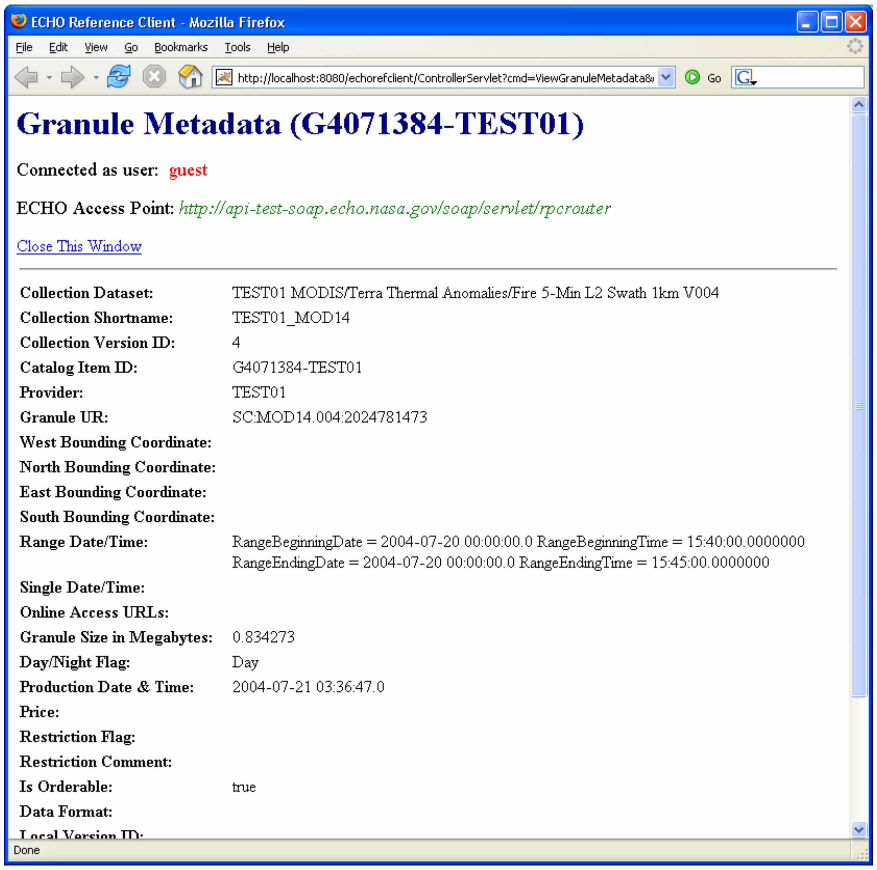
Granule metadata can be searched in ECHO using constraints on geographical spatial extent specified in a variety of formats. Section 6.1.3 of the ECHO User's Guide provides a full description of the spatial search capabilities and formats.

The spatial search implemented in the Query Granules function of the Reference Client uses the ECHO `<IIMSPolygon>` element to define a bounding rectangle for the upper-left and lower-right coordinates specified by the user. For an Upper-Left Coordinate of Latitude=41.0 and Longitude=-110.0, and a Lower-Right Coordinate of Latitude=37.0 and Longitude=-102.0, the message sent to ECHO will contain the following Granule spatial condition:

```
<granuleCondition>
  <spatial>
    <IIMSPolygon>
      <IIMSLRing>
        <IIMSPoint long='-102.0' lat='37.0'/>
        <IIMSPoint long='-102.0' lat='41.0'/>
        <IIMSPoint long='-110.0' lat='41.0'/>
        <IIMSPoint long='-110.0' lat='37.0'/>
        <IIMSPoint long='-102.0' lat='37.0'/>
      </IIMSLRing>
    </IIMSPolygon>
  </spatial>
</granuleCondition>
```

Note that the point coordinates are provided in counter-clockwise order. This is a requirement of ECHO's underlying Oracle database server.

Click on the *ECHO Item ID* of one of the Granules. A new window is displayed with detailed information about the Granule (Figure 18).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=ViewGranuleMetadata&
Go
Done

Granule Metadata (G4071384-TEST01)

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Close This Window

Collection Dataset:	TEST01 MODIS/Terra Thermal Anomalies/Fire 5-Min L2 Swath 1km V004
Collection Shortname:	TEST01_MOD14
Collection Version ID:	4
Catalog Item ID:	G4071384-TEST01
Provider:	TEST01
Granule UR:	SC:MOD14.004:2024781473
West Bounding Coordinate:	
North Bounding Coordinate:	
East Bounding Coordinate:	
South Bounding Coordinate:	
Range Date/Time:	RangeBeginningDate = 2004-07-20 00:00:00.0 RangeBeginningTime = 15:40:00.0000000 RangeEndingDate = 2004-07-20 00:00:00.0 RangeEndingTime = 15:45:00.0000000
Single Date/Time:	
Online Access URLs:	
Granule Size in Megabytes:	0.834273
Day/Night Flag:	Day
Production Date & Time:	2004-07-21 03:36:47.0
Price:	
Restriction Flag:	
Restriction Comment:	
Is Orderable:	true
Data Format:	
Local Version ID:	

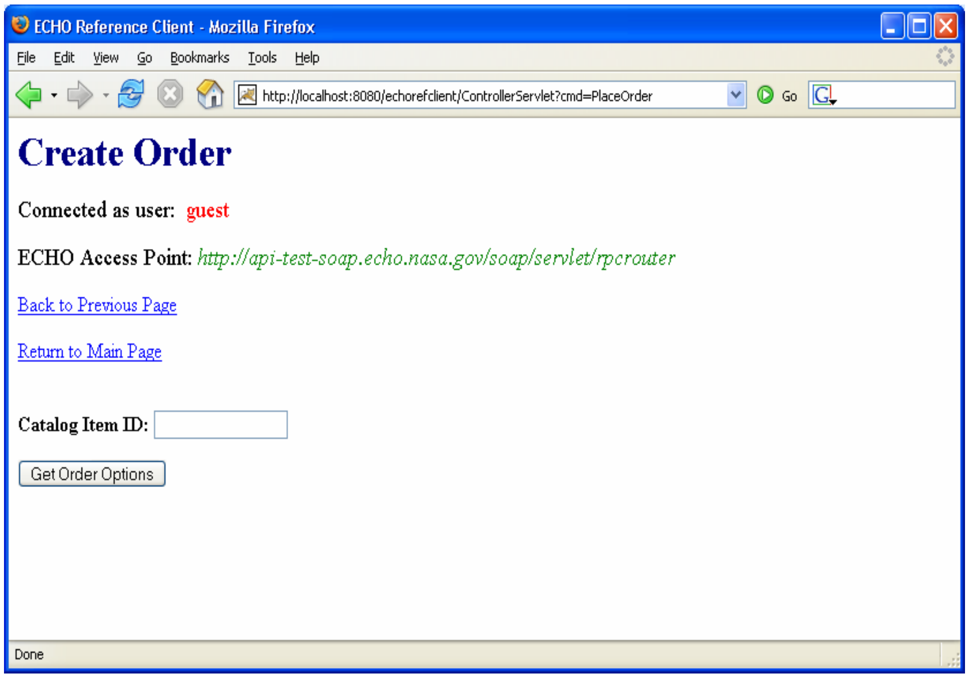
Figure 18 – Granule Metadata Page

<u>Granule Metadata</u>

In the ECHO data model Granules are made up of dozens of metadata fields defined in the http://api.echo.nasa.gov/echo/dtd/ECHOGranuleResults.dtd. The Reference Client only captures a small subset of these fields.

3.7 Order Items

So far we've learned about the Data Providers available in the system, searched for a set of Collections and Granules, and viewed Collection and Granule metadata. The last function of the Reference Client is the ordering of the data. Similar to querying for Granules directly from the Collection results, we can place an order for an item (Collection or Granule) if an "order" link is available in the results page (Figure 17). We can also order an item by entering the ECHO item ID directly in the Order Page (Figure 19).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=PlaceOrder
Go
Done

Create Order

Connected as user: guest

ECHO Access Point: http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter

Back to Previous Page

Return to Main Page

Catalog Item ID:

Get Order Options

Figure 19 - Order Page

<u>Order</u>

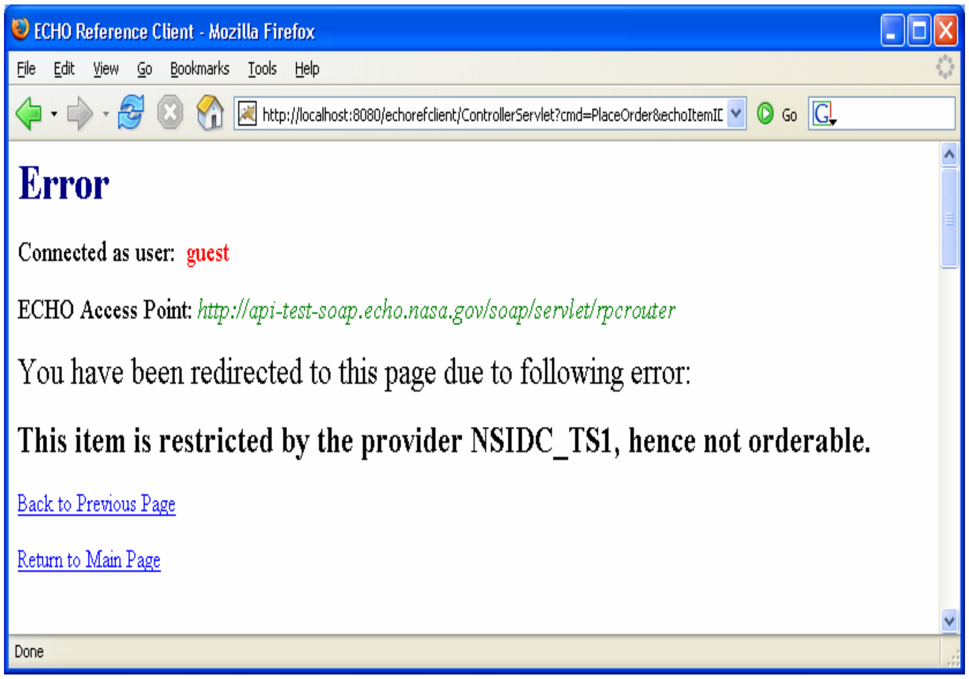
The *Order* function makes use of:

- ECHO API Order Entry Service
    - Present Catalog Item Transaction
    - Create Order Transaction
    - Set User Information For Order Transaction
    - Validate Transaction
    - Submit Order Transaction

The Reference Client only allows ordering of one item at a time. The ECHO API allows orders with multiple items (and items from multiple providers) to be created and submitted together.

You should still have your Query Granule Results page (Figure 17) open. Click on one of the Order links next to any of the items.

<u>Please note</u> that some items may be restricted from ordering by guest or registered users. When this occurs, an error will result from the Reference Client's attempt to retrieve order options and begin the order process. Figure 20 shows the error page that is displayed to the user.


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=PlaceOrder&echoItemIE
Go
Done

Error

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

You have been redirected to this page due to following error:

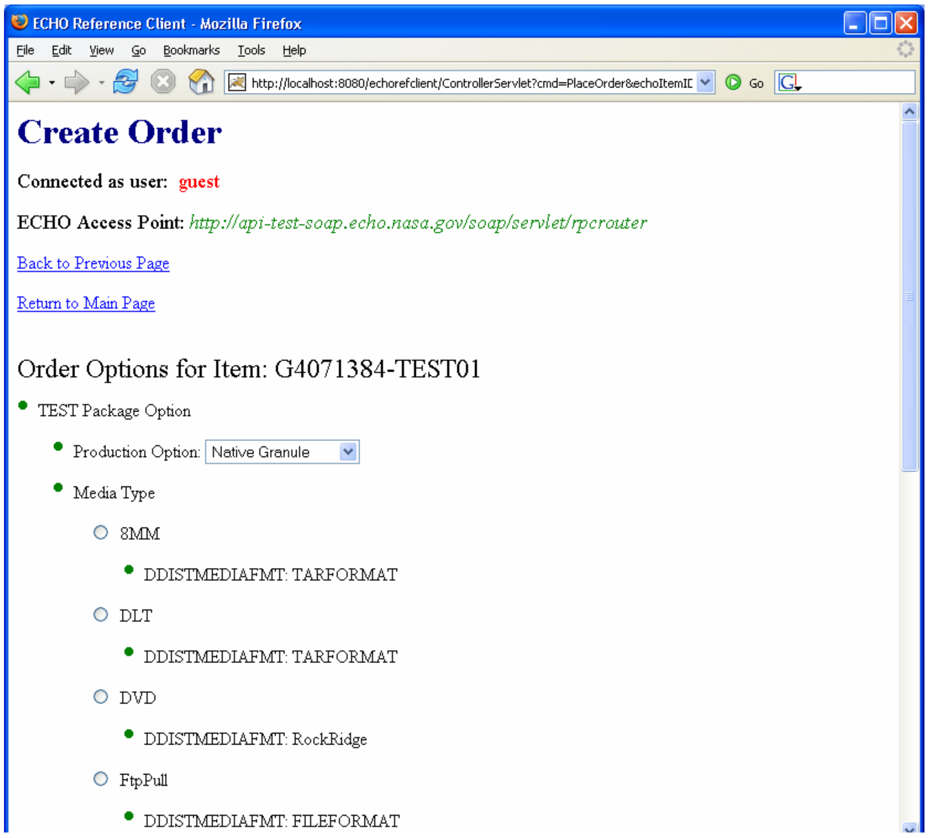
This item is restricted by the provider NSIDC_TS1, hence not orderable.

Back to Previous Page

Return to Main Page

Figure 20 – Error that results when user attempts to order item that is restricted.

If the item selected is available for order by the user, the Order page is displayed (Figure 21).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=PlaceOrder&echoItemIE
Go

Create Order

Connected as user: guest

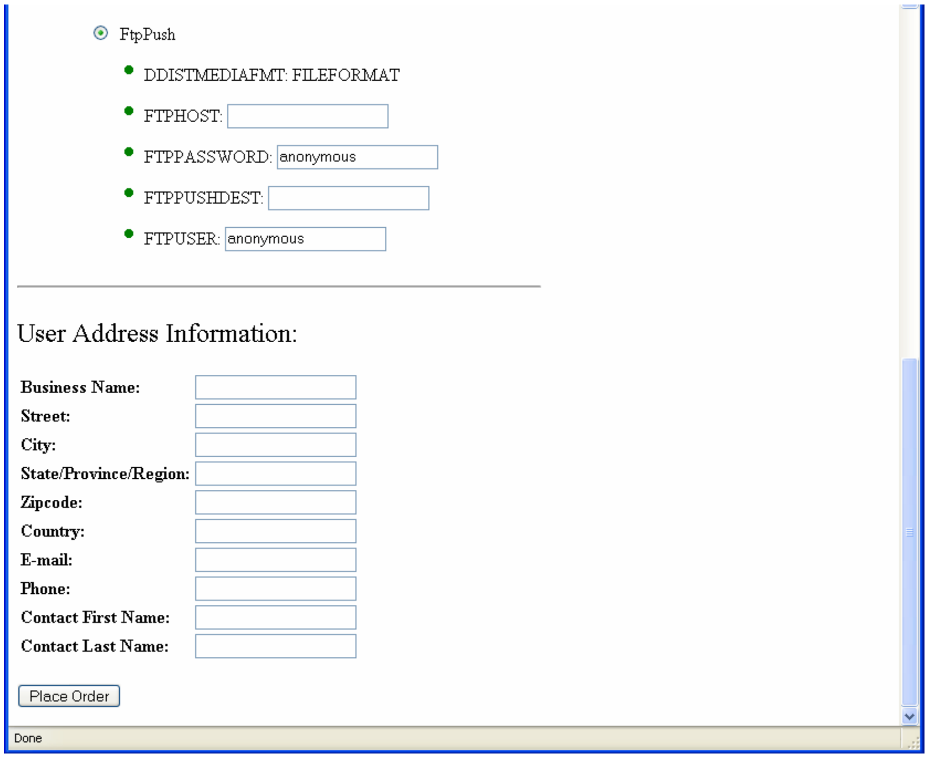
ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Back to Previous Page

Return to Main Page

Order Options for Item: G4071384-TEST01

- TEST Package Option
  - Production Option: Native Granule
  - Media Type
    - ○ 8MM
      - DDISTMEDIAFMT: TARFORMAT
    - ○ DLT
      - DDISTMEDIAFMT: TARFORMAT
    - ○ DVD
      - DDISTMEDIAFMT: RockRidge
    - ○ FtpPull
      - DDISTMEDIAFMT: FILEFORMAT



[✓ FtpPush]

- DDISTMEDIAFMT: FILEFORMAT
- FTPHOST:
- FTPPASSWORD: anonymous
- FTPPUSHDEST:
- FTPUSER: anonymous

User Address Information:

Business Name:	
Street:	
City:	
State/Province/Region:	
Zipcode:	
Country:	
E-mail:	
Phone:	
Contact First Name:	
Contact Last Name:	

Figure 21 - Create Order Page (shown over two browser screenshots)

The *Create Order Page* is composed of two parts: a) the item order options, and b) the user information form. You must fill both sections to create an order. First complete the order parameters by selecting the choices given and filling the necessary fields with values. Note that the text fields contain the default values for the parameter.

Order Options

The Order Option Selection tree is not standardized across ECHO. Each Data Provider specifies a set of *complex* and *simple* options for the orderable items from their catalog. The Reference Client obtains these options and displays them to the user in the form of an option tree.

A *complex* option is a grouping of values (either a *structure* or a *choice*) composed of other *complex* options and/or *simple* options. *Simple* options may consist of a single primitive value or may require the selection of values from a set.

Next fill the bottom part of the page by entering address information for the order.

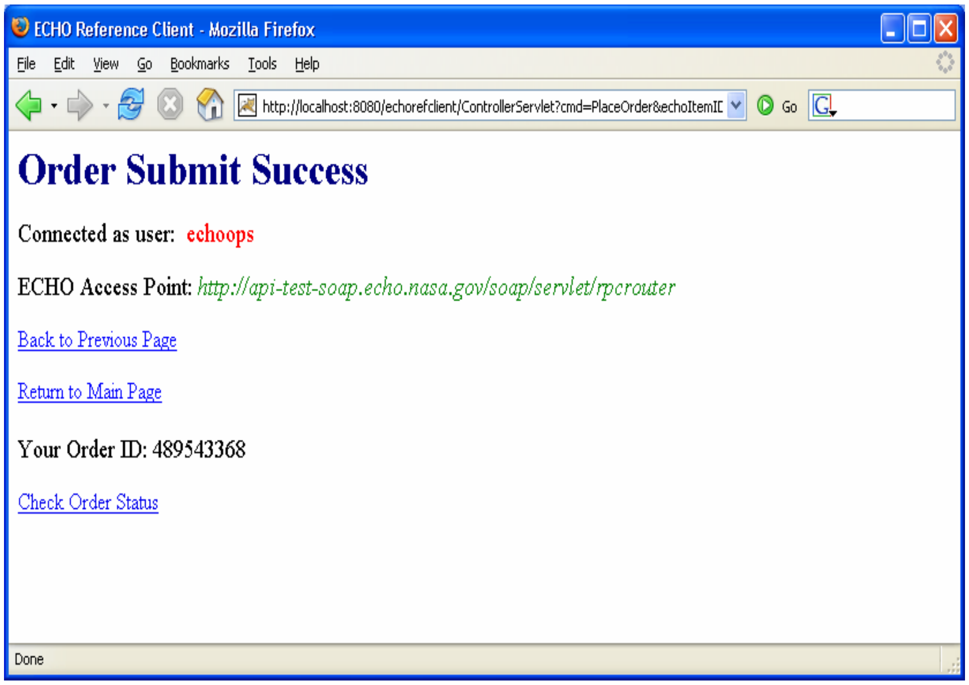
<u>Order Address</u>

ECHO allows three addresses (shipping, billing, and contact) to be entered for an order. The Reference Client only has the user enter <u>one</u> address and uses this information to populate all three addresses in the order that is sent to ECHO. Furthermore, the Client does not perform any validation of the values entered for the address. Be sure to enter valid values, particularly:

- Zipcode – 5 digit value
- Country – three-digit ISO country abbreviation (e.g., "USA")

Please note that although ECHO provides "address book" capabilities for registered users, the ability to access this information and use it in creating orders was not implemented in this version of the Reference Client.

Now click the *Place Order* button and submit the order. The order result page is displayed along with an ECHO order ID (Figure 22).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=PlaceOrder&echoItemIL
Go
Done

Order Submit Success

Connected as user: echoops

ECHO Access Point: http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter

Back to Previous Page

Return to Main Page

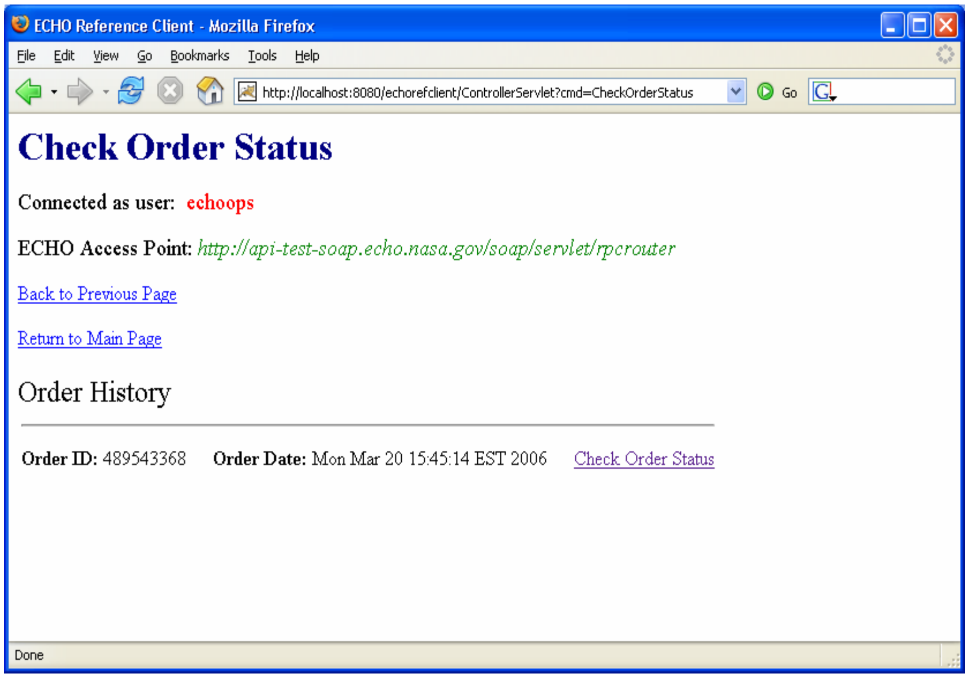
Your Order ID: 489543368

Check Order Status

Figure 22 – Order Submit Success Page

3.8 Check Order Status

Finally, now that you have the order ID, you can check the order status by selecting the *Check Order Status* link from the Main page (Figure 8). If you're connected as a registered user the *Check Order Status Page* will appear displaying your order history which includes the order ID and date of all previously placed orders (Figure 23).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=CheckOrderStatus
Go
Done

Check Order Status

Connected as user: echoops

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Back to Previous Page

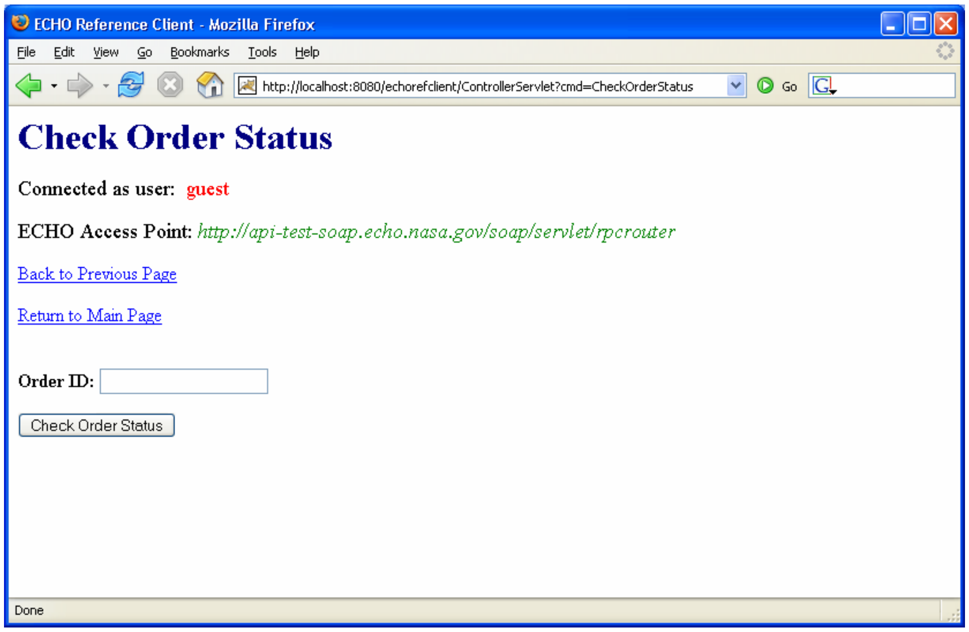
Return to Main Page

Order History

Order ID: 489543368 **Order Date:** Mon Mar 20 15:45:14 EST 2006 Check Order Status

Figure 23 - Check Order Status Page for Registered Users

Otherwise, if you're acting as a Guest user, the page will instead provide a form with a field for manually entering an ECHO Order ID (Figure 24).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=CheckOrderStatus
Go
Check Order Status
Done

Check Order Status

Connected as user: guest

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Back to Previous Page

Return to Main Page

Order ID:

Figure 24 - Check Order Status Page for Non-registered Users

Order History

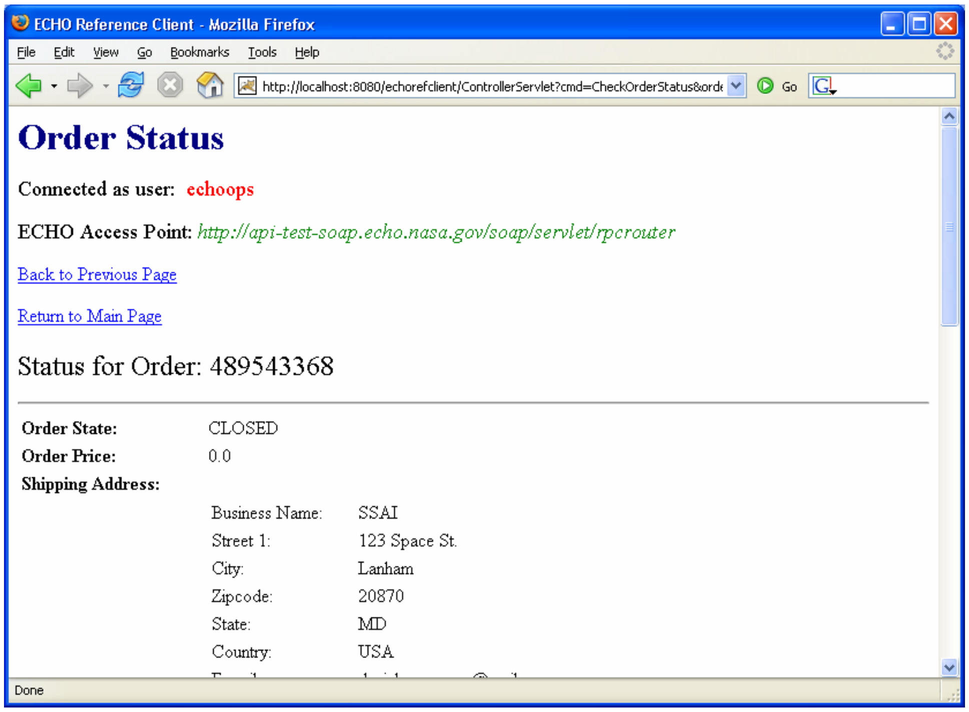
The *Check Order Status* function makes use of:

- ECHO API User Account Service

 - Present Order History Transaction
- ECHO API Order Entry Service
  - Present Order Transaction

Note that a listing of the previously placed orders is only available for registered accounts. Guest users will have to record their Order IDs externally and use this information to subsequently check the order status.

To view the details for a particular order, select the *Check Order Status* link next to the particular order, or enter an ECHO Order ID and select the *Check Order Status* button. The *Order Status Page* will be displayed with details about the order (Figure 25).


ECHO Reference Client - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/echorefclient/ControllerServlet?cmd=CheckOrderStatus&orde
Go
Done

Order Status

Connected as user: echoops

ECHO Access Point: *http://api-test-soap.echo.nasa.gov/soap/servlet/rpcrouter*

Back to Previous Page

Return to Main Page

Status for Order: 489543368

Order State:	CLOSED	
Order Price:	0.0	
Shipping Address:		
	Business Name:	SSAI
	Street 1:	123 Space St.
	City:	Lanham
	Zipcode:	20870
	State:	MD
	Country:	USA

Figure 25 - Order Status Page

3.9 The End of the Tour

Let's finally disconnect the client from ECHO. Return to the Main page and click on the *Disconnect* link. The client will refresh the Main page and the *Disconnect* link will have been replaced by the *Connect* link.

This concludes our guided tour of the ECHO Reference Client. It should have given you a good overview of some core ECHO services and how they may be interfaced in a simple web application. The next section of this document provides information on the design of the software.

4 Inner Workings

This section gives an overview of the Reference Client and exposes the high-level design of the application through a series of UML diagrams. Details of the actual implementation are left to the comments found in the source code and the *Javadoc* documentation that accompanies it (refer to the `/echorefclient-X.X/doc/api` directory).

4.1 Components

The Reference Client is essentially divided into two subsystems: the web user interface and the ECHO communication library. The former handles all user interaction and directs the flow of events between the various interface components. The latter is an object-oriented abstraction of the ECHO Services, transactions, and data model.

The web interface uses a simple Model-View-Controller design by having a main *Controller Servlet* that receives all of the incoming web requests and then forwards them to the appropriate *Command* class (Figure 26). Each Command, in turn, performs a set of internal actions (usually interacting directly with the ECHO server) and then redirects the user to a JavaServer Page (JSP) that will display the appropriate user interface.

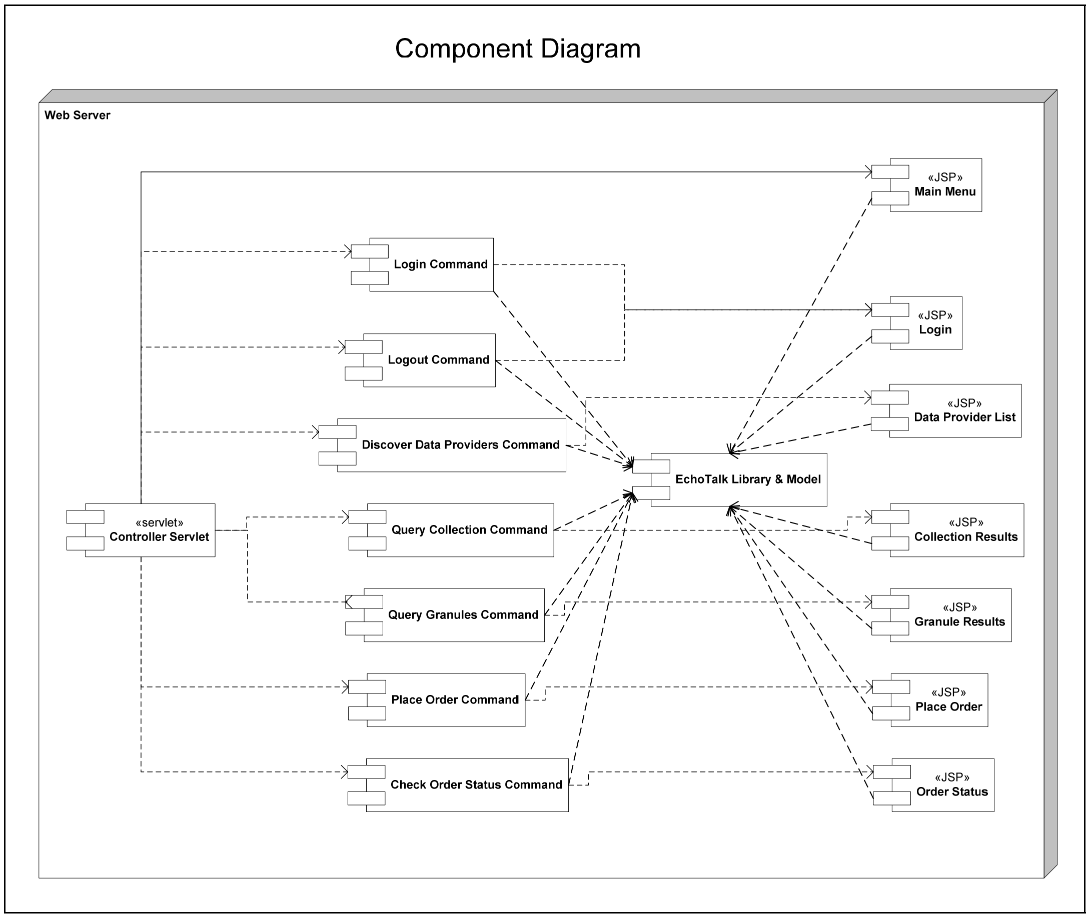


Figure 26 - Component Diagram

The *EchoTalk* library is the module that directly interacts with ECHO. It is an object-oriented abstraction of the ECHO Services, transactions, and data model. Only a subset of the ECHO API has been implemented:

- Catalog Service
  - Present
  - Query

- Provider Profile Service
  - List All Providers
  - Present Provider Profile

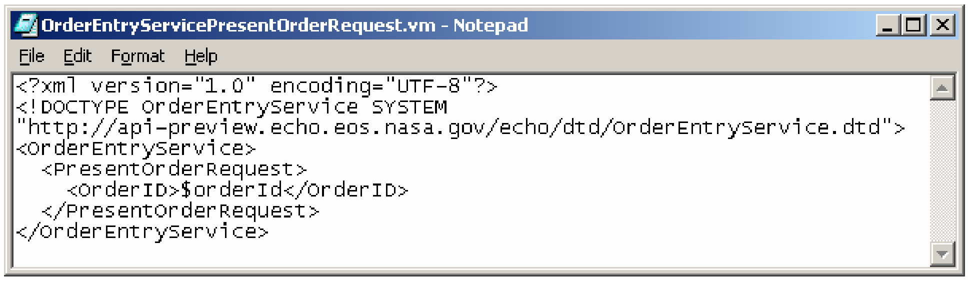
- Order Service
  - Present Catalog Item
  - Create Order
  - Validate Order
  - Submit Order
  - Present Order

- User Account Service
  - Add Address Information
  - Add Phone Information
  - Change User Password
  - Delete Address Information
  - Delete Phone Information
  - Present Order History
  - Update Address Information
  - Update Phone Information
  - Update User Information
  - Recall Username
  - Reset User Password

EchoTalk is essentially made up of *model*, *parser*, and *service* classes. The *model* classes represent various ECHO entities such as Collections, Granules, Orders, Addresses, Order Options, and more. The *parser* classes build *model* classes from the XML messages returned from ECHO. The *service* classes assemble transaction XML message requests to be sent to ECHO.

Internally, the assembly of the XML is done with the use of the open source *Apache Jakarta Velocity* templating engine (http://jakarta.apache.org/velocity). The XML documents are not created programmatically, but instead come from a repository of textual templates with predefined locations in the text to be replaced with actual values. All of the templates get stored in the application's Classpath and can be found in the distribution `source/templates` directory.

As an example, consider the template for the Order Entry Service *Present Order* transaction (`source/templates/OrderEntryServiceParserOrderRequest.vm`--Figure 27). There is a placeholder for the Order ID in the form of the variable `$orderId`. This variable will get replaced with an actual value by the Velocity rendering engine before the document is sent to ECHO.



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE OrderEntryService SYSTEM
"http://api-preview.echo.eos.nasa.gov/echo/dtd/OrderEntryService.dtd">
<OrderEntryService>
  <PresentOrderRequest>
    <OrderID>$orderId</OrderID>
  </PresentOrderRequest>
</OrderEntryService>
```

Figure 27 - Example Velocity Template

Using templates makes it very easy to modify and process the XML requests to be sent to ECHO.

4.2 Sample Interactions

A set of sequence diagrams are provided here to demonstrate the Reference Client in action. These diagrams show the interactions that occur to and from: the web browser, the user interface, and the ECHO server.

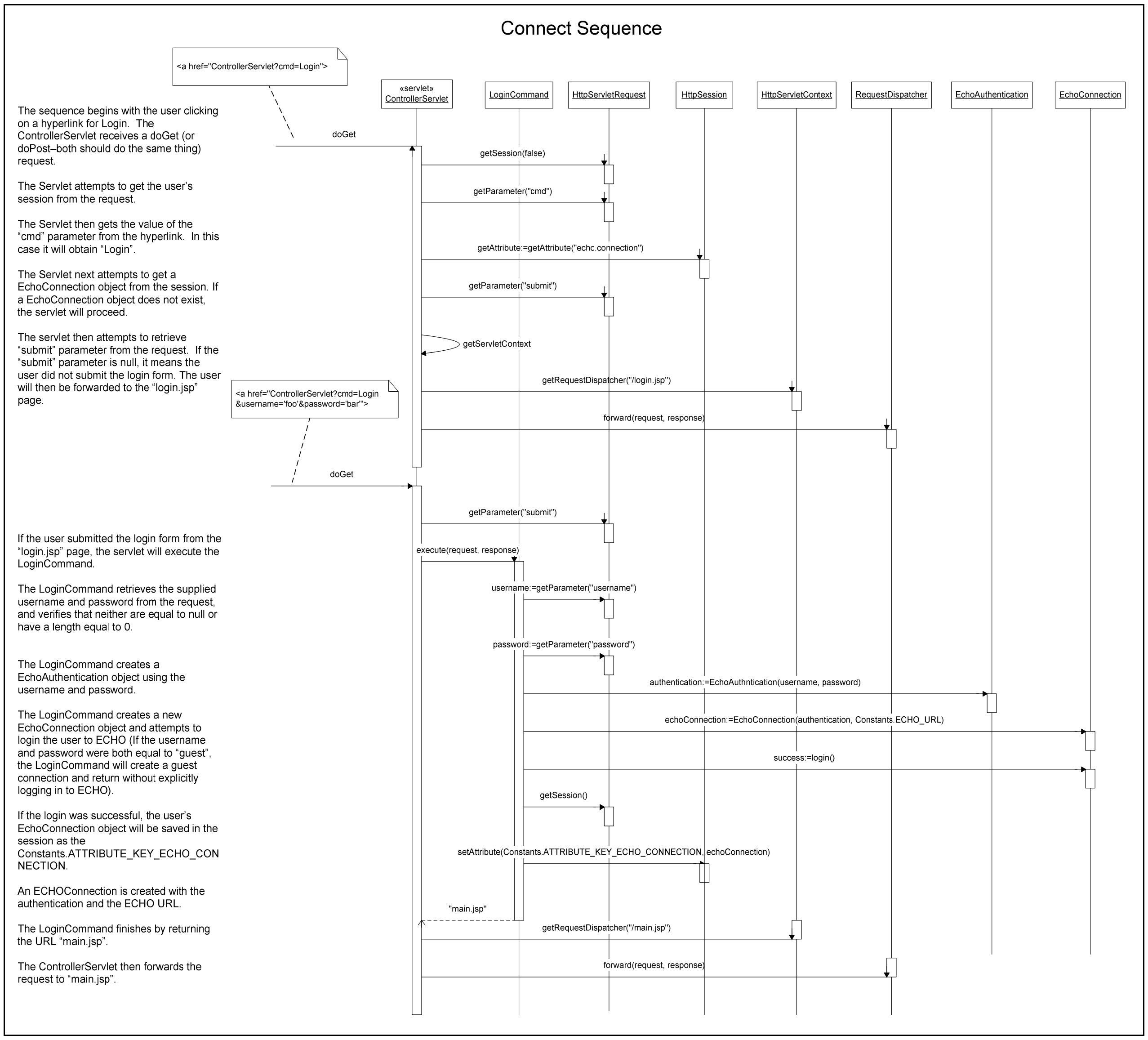
4.2.1 Connect Sequence

This sequence begins with the `ControllerServlet` receiving the request to connect. The Servlet analyzes the HTTP parameters and places pieces of information (in the form of Java objects) in the Request and Session. It then forwards the user to the Connect JSP page, which displays an HTML form for the user to fill with login parameters (i.e., credentials—username and password).

When the user submits the form, the `ControllerServlet` is called again. The Servlet now determines that it is coming from the login form and forwards the execution to the `LoginCommand` class.

The Login Command will now examine the Request and Session objects and pull out the needed parameters. The Command then creates an instance of the `EchoAuthentication` class with the user's credentials. The authentication is then passed on to the `EchoConnection` object which performs login and establishes a new session with ECHO.

This general pattern is repeated for all of the Reference Client functions.


Connect Sequence
<a href="ControllerServlet?cmd=Login">
«servlet» ControllerServlet
LoginCommand
HttpServletRequest
HttpSession
HttpServletContext
RequestDispatcher
EchoAuthentication
EchoConnection
doGet
getSession(false)
getParameter("cmd")
getAttribute:=getAttribute("echo.connection")
getParameter("submit")
getServletContext
getRequestDispatcher("/login.jsp")
<a href="ControllerServlet?cmd=Login &username='foo'&password='bar'">
forward(request, response)
doGet
getParameter("submit")
execute(request, response)
username:=getParameter("username")
password:=getParameter("password")
authentication:=EchoAuthntication(username, password)
echoConnection:=EchoConnection(authentication, Constants.ECHO_URL)
success:=login()
getSession()
setAttribute(Constants.ATTRIBUTE_KEY_ECHO_CONNECTION, echoConnection)
"main.jsp"
getRequestDispatcher("/main.jsp")
forward(request, response)

The sequence begins with the user clicking on a hyperlink for Login. The ControllerServlet receives a doGet (or doPost–both should do the same thing) request.

The Servlet attempts to get the user's session from the request.

The Servlet then gets the value of the "cmd" parameter from the hyperlink. In this case it will obtain "Login".

The Servlet next attempts to get a EchoConnection object from the session. If a EchoConnection object does not exist, the servlet will proceed.

The servlet then attempts to retrieve "submit" parameter from the request. If the "submit" parameter is null, it means the user did not submit the login form. The user will then be forwarded to the "login.jsp" page.

If the user submitted the login form from the "login.jsp" page, the servlet will execute the LoginCommand.

The LoginCommand retrieves the supplied username and password from the request, and verifies that neither are equal to null or have a length equal to 0.

The LoginCommand creates a EchoAuthentication object using the username and password.

The LoginCommand creates a new EchoConnection object and attempts to login the user to ECHO (If the username and password were both equal to "guest", the LoginCommand will create a guest connection and return without explicitly logging in to ECHO).

If the login was successful, the user's EchoConnection object will be saved in the session as the Constants.ATTRIBUTE_KEY_ECHO_CONNECTION.

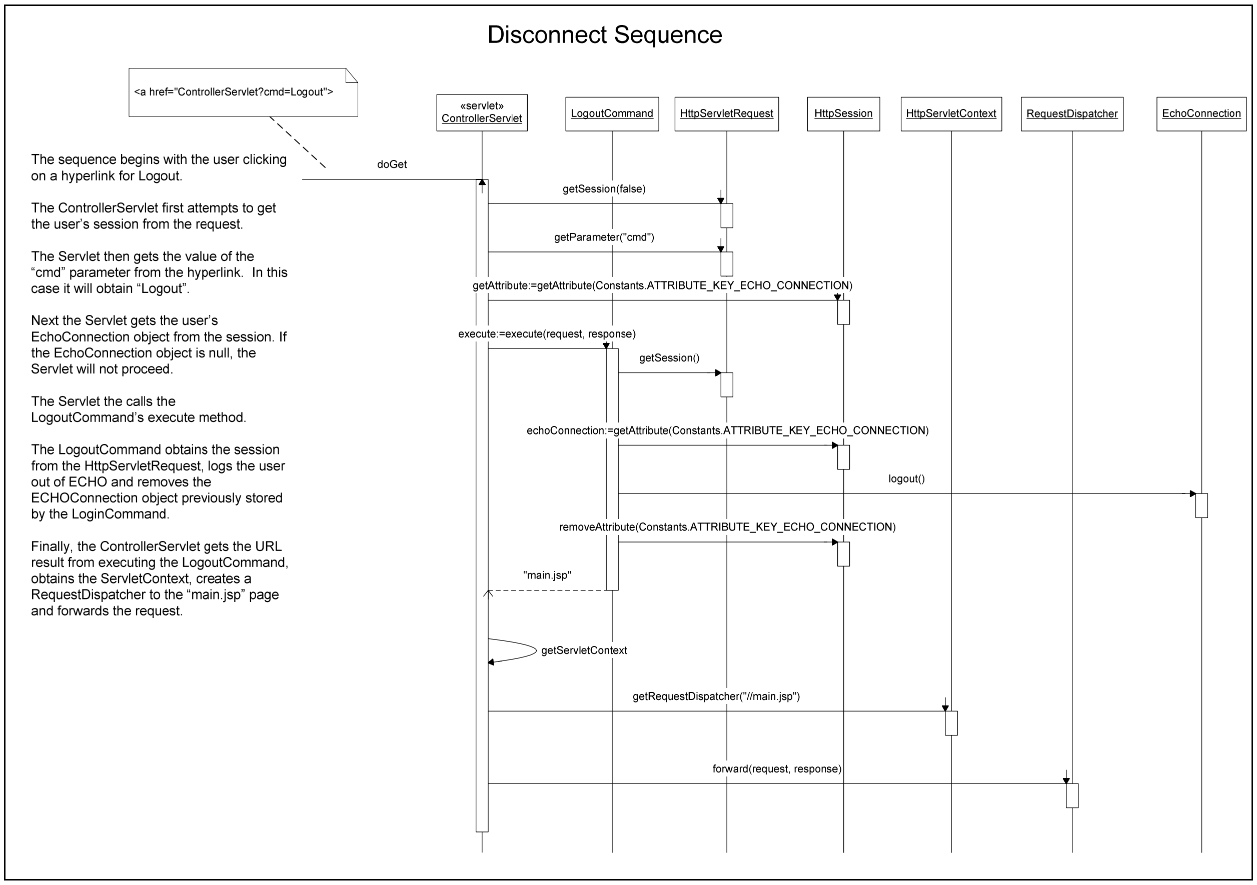
An ECHOConnection is created with the authentication and the ECHO URL.

The LoginCommand finishes by returning the URL "main.jsp".

The ControllerServlet then forwards the request to "main.jsp".

Figure 28 - Connect Sequence Diagram

4.2.2 Disconnect Sequence



The sequence begins with the user clicking on a hyperlink for Logout.

The ControllerServlet first attempts to get the user's session from the request.

The Servlet then gets the value of the "cmd" parameter from the hyperlink. In this case it will obtain "Logout".

Next the Servlet gets the user's EchoConnection object from the session. If the EchoConnection object is null, the Servlet will not proceed.

The Servlet the calls the LogoutCommand's execute method.

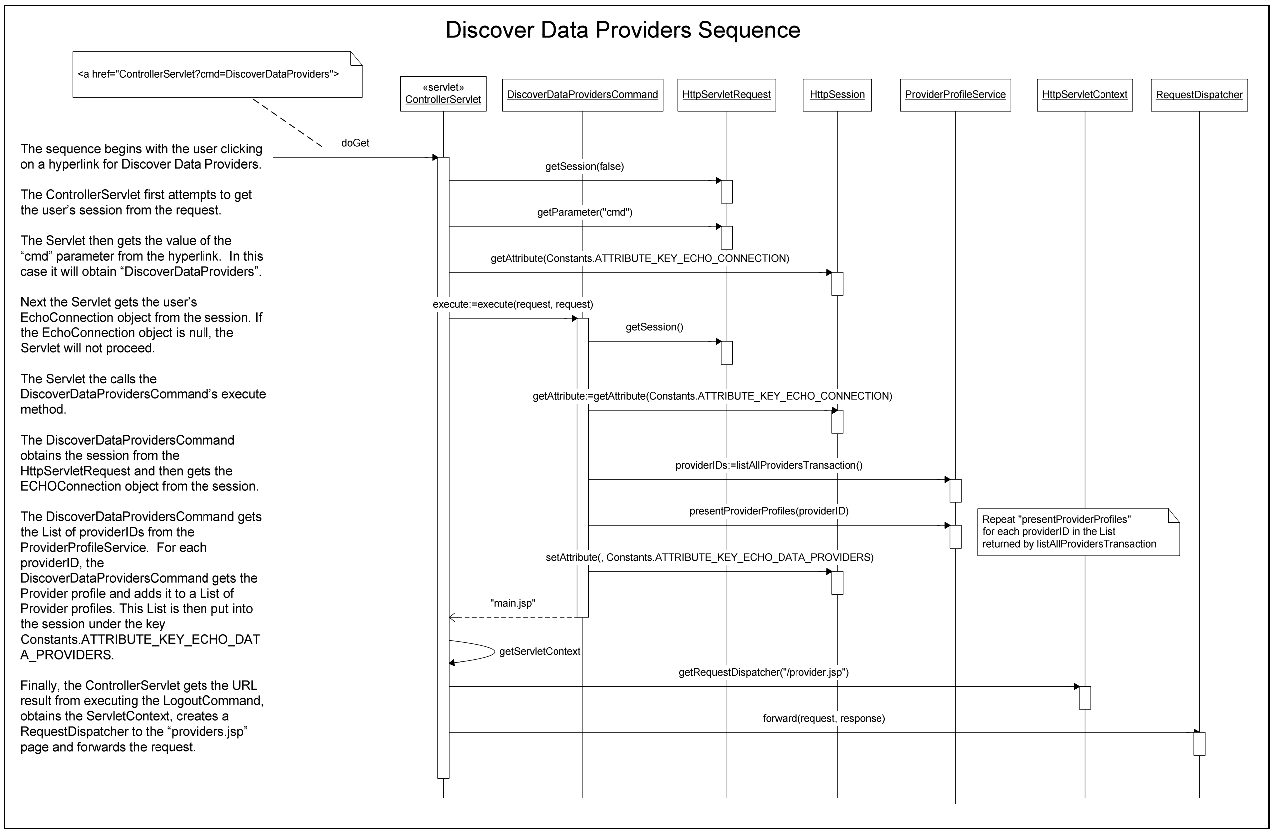
The LogoutCommand obtains the session from the HttpServletRequest, logs the user out of ECHO and removes the ECHOConnection object previously stored by the LoginCommand.

Finally, the ControllerServlet gets the URL result from executing the LogoutCommand, obtains the ServletContext, creates a RequestDispatcher to the "main.jsp" page and forwards the request.

Disconnect Sequence
<a href="ControllerServlet?cmd=Logout">
«servlet» ControllerServlet
LogoutCommand
HttpServletRequest
HttpSession
HttpServletContext
RequestDispatcher
EchoConnection
doGet
getSession(false)
getParameter("cmd")
getAttribute:=getAttribute(Constants.ATTRIBUTE_KEY_ECHO_CONNECTION)
execute:=execute(request, response)
getSession()
echoConnection:=getAttribute(Constants.ATTRIBUTE_KEY_ECHO_CONNECTION)
logout()
removeAttribute(Constants.ATTRIBUTE_KEY_ECHO_CONNECTION)
"main.jsp"
getServletContext
getRequestDispatcher("//main.jsp")
forward(request, response)

Figure 29 - Disconnect Sequence Diagram

4.2.3 Discover Data Providers Sequence


Discover Data Providers Sequence
<a href="ControllerServlet?cmd=DiscoverDataProviders">
«servlet» ControllerServlet
DiscoverDataProvidersCommand
HttpServletRequest
HttpSession
ProviderProfileService
HttpServletContext
RequestDispatcher
doGet
getSession(false)
getParameter("cmd")
getAttribute(Constants.ATTRIBUTE_KEY_ECHO_CONNECTION)
execute:=execute(request, request)
getSession()
getAttribute:=getAttribute(Constants.ATTRIBUTE_KEY_ECHO_CONNECTION)
providerIDs:=listAllProvidersTransaction()
presentProviderProfiles(providerID)
Repeat "presentProviderProfiles" for each providerID in the List returned by listAllProvidersTransaction
setAttribute(, Constants.ATTRIBUTE_KEY_ECHO_DATA_PROVIDERS)
"main.jsp"
getServletContext
getRequestDispatcher("/provider.jsp")
forward(request, response)

The sequence begins with the user clicking on a hyperlink for Discover Data Providers.

The ControllerServlet first attempts to get the user's session from the request.

The Servlet then gets the value of the "cmd" parameter from the hyperlink. In this case it will obtain "DiscoverDataProviders".

Next the Servlet gets the user's EchoConnection object from the session. If the EchoConnection object is null, the Servlet will not proceed.

The Servlet the calls the DiscoverDataProvidersCommand's execute method.

The DiscoverDataProvidersCommand obtains the session from the HttpServletRequest and then gets the ECHOConnection object from the session.

The DiscoverDataProvidersCommand gets the List of providerIDs from the ProviderProfileService. For each providerID, the DiscoverDataProvidersCommand gets the Provider profile and adds it to a List of Provider profiles. This List is then put into the session under the key Constants.ATTRIBUTE_KEY_ECHO_DAT A_PROVIDERS.

Finally, the ControllerServlet gets the URL result from executing the LogoutCommand, obtains the ServletContext, creates a RequestDispatcher to the "providers.jsp" page and forwards the request.

Figure 30 – Discover Data Providers Sequence Diagram

4.2.4 Provider Profile Sequence

This sequence shows the EchoTalk subsystem at work. The `ProviderProfileService` class will use a separate parser and an ECHO connection object to communicate with ECHO. Along the way the Jakarta Velocity templating library will be used to render the XML requests to ECHO, as previously described in Section 4.1.

Provider Profile Service Sequence

A new ProviderProfileService is instantiated. The constructor takes an EchoConnection as parameter. It will use the credentials associated with that connection to establish a session with ECHO for all transactions.

A timeout value is set for all of the transactions executed by the service.

The listAllProvidersTransaction is called with no parameters. It, in turn, calls the internal method performTransaction with the name of the template to use for the XML message to be sent to ECHO for this transaction.

The performTransaction creates and starts a new ECHORequestThread. The method then waits for that thread to join back.

The ECHORequestThread initializes the Velocity engine and renders the XML template.

The rendered template is then sent to ECHO via the ECHOConnection object associated with the ProviderProfileService.

When the response is received from ECHO it is then sent to the ProviderProfileServiceParser to be parsed for any errors. If no errors are detected, the XML response is then returned to performTransaction which then sends it back to the listAllProvidersTransaction method.

Now that XML response is sent to the ProviderProfileServiceParser to be parsed. The Provider IDs are extracted and returned as a list of String objects.

This same pattern is repeated when the presentProviderProfileRequest is called,

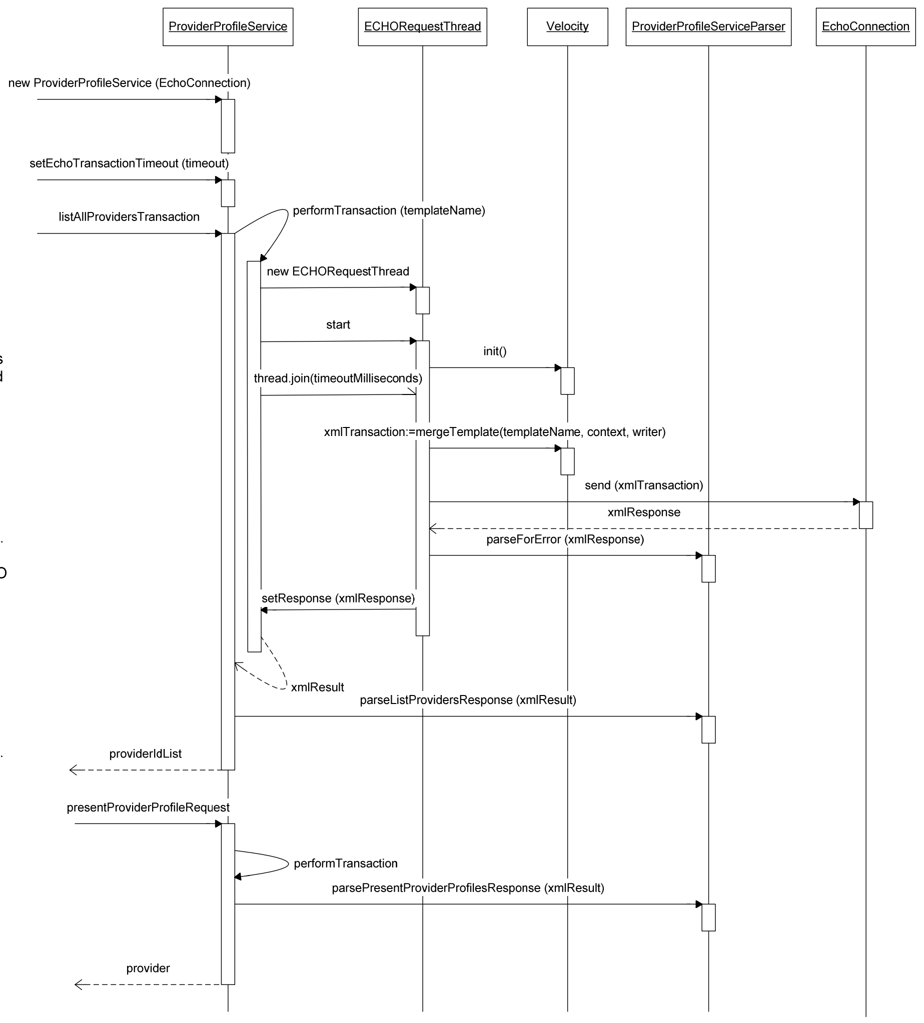

ProviderProfileService
ECHORequestThread
Velocity
ProviderProfileServiceParser
EchoConnection
new ProviderProfileService (EchoConnection)
setEchoTransactionTimeout (timeout)
listAllProvidersTransaction
performTransaction (templateName)
new ECHORequestThread
start
init()
thread.join(timeoutMilliseconds)
xmlTransaction:=mergeTemplate(templateName, context, writer)
send (xmlTransaction)
xmlResponse
parseForError (xmlResponse)
setResponse (xmlResponse)
xmlResult
parseListProvidersResponse (xmlResult)
providerIdList
presentProviderProfileRequest
performTransaction
parsePresentProviderProfilesResponse (xmlResult)
provider

Figure 31 – Provider Profile Sequence Diagram

5 Developing Your Client

As mentioned in the introduction, the ECHO Reference Client is a learning tool. While it is a functional ECHO client, it lacks the robustness and user interface elements that would be required for an "operational" ECHO client. When you develop your ECHO client, consider some of these suggestions:

- **Use an internal cache for frequently accessed data –** information such as the list of Data Providers and their profiles could be reused instead of being constantly fetched from ECHO every time they are requested.

- **Use drop-down lists in the UI –** the user interface could provide drop-down lists or checkboxes for several fields (e.g. Data Center IDs, Dataset IDs, Sensor names) in the search forms instead of requiring users to provide text input. You can use the Provider Profile Service *ListAllProvidersRequest* transaction to get a list of valid Provider IDs (same as Data Center IDs) and Catalog Service transactions to extract values for other relevant fields.

- **Implement a mechanism to prevent multiple repeated requests –** repeated submission of the same ECHO requests should be prevented at the user interface. If a user sends a search request the interface should make the user wait for the results. Otherwise a user may flood ECHO with repeated requests.

- **Expose more relevant search parameters –** additional search criteria could be added to the Collection and Granule query forms to make the client more meaningful to the client user community.

- **Obtain user information from the User Account Service –** a registered user stores his profile in the ECHO server. This profile could be queried to extract the user's address information for the Order Item function.

We encourage you to examine, modify and use the source code of the Reference Client to the extent that it may help you write your own ECHO client. But before diving right into the source code, we recommend that you generate the API documentation (see Section 2.3) and read the class descriptions, particularly for the "Command" classes and the EchoTalk packages. These provide a summary of the class' purpose and functionality and help you identify those components relevant to your own client development.